

Resource-Competitive Communication

Valerie King*

Jared Saia†

Maxwell Young‡

March 1, 2012

Abstract

Consider the general scenario where Alice wishes to transmit a message m to Bob. These two players share a communication channel; however, there exists an adversary, Carol, who aims to prevent the transmission of m by blocking this channel. There are costs to send, receive or block m on the channel, and we ask: How much do Alice and Bob need to spend relative to the adversary Carol in order to guarantee transmission of m ?

This problem abstracts many types of conflict in information networks including: jamming attacks in wireless sensor networks (WSNs) and distributed denial-of-service (DDoS) attacks on the Internet, where the costs to Alice, Bob and Carol represent an expenditure of energy or network resources. The problem allows us to quantitatively analyze the economics of information exchange in an adversarial setting and ask: Is communication cheaper than censoring communication?

We answer this question in the affirmative. Specifically, in a time-slotted network with constant costs to send, receive and block m in a slot, if Carol spends a total of B slots trying to block m , then both Alice and Bob must be active for only $O(B^\varphi + 1) = O(B^{.62} + 1)$ slots in expectation to transmit m , where $\varphi = (1 + \sqrt{5})/2$ is the golden ratio. Surprisingly, this result holds even if (1) B is *unknown* to either player; (2) Carol knows the algorithms of both players, but not their random bits; and (3) Carol can attack using total knowledge of past actions of both players.

In the spirit of competitive analysis, approximation guarantees, and game-theoretic treatments, our approach represents another notion of relative performance that we call *resource competitiveness*. This new metric measures the worst-case performance of an algorithm relative to *any* adversarial strategy and pertains to scenarios where all network devices are resource-constrained. Here, we apply the resource-competitive results above to two concrete problems. First, we consider jamming attacks in WSNs and address the fundamental task of propagating m from a single device to all others in the presence of faults. Second, we examine how to mitigate application-level DDoS attacks in a wired client-server scenario.

1 Introduction

Denial-of-service (DoS) attacks are an increasingly common threat to the availability of large-scale networks. In the wireless domain, practitioners have expressed concerns that important surveillance services will suffer DoS attacks by the deliberate disruption of the communication medium [16] and there is a large body of literature on various practical security concerns (see [48, 54] and references therein). In the wired domain, the distributed denial-of-service (DDoS) attacks in November of 2010 on Amazon, Visa, Mastercard, and PayPal [9, 44, 68] demonstrated that even prominent companies are vulnerable over the Internet. Given this state of affairs, an interesting question arises: Is it fundamentally “harder” or “easier” to communicate in such large-scale networks than it is to block communication?

*Department of Computer Science, University of Victoria, BC, Canada; email: val@cs.uvic.ca

†Contact Author: Department of Computer Science, University of New Mexico, NM, USA; email: saia@cs.unm.edu.

‡Department of Computer Science, National University of Singapore, Singapore; email: dcsmmry@nus.edu.sg

To analyze this question from an algorithmic perspective, we define the following simple problem, which we call the *3-Player Scenario*: Alice wishes to guarantee transmission of a message m directly to Bob over a single communication channel. However, there exists an adversary Carol who aims to prevent communication by blocking transmissions over the channel. We consider two cases: (Case 1) when Carol may spoof or even control Bob, which allows her to manipulate an unwitting Alice into incurring excessive sending costs; and (Case 2) where Bob is both correct and unspoofable, and his communications cannot be blocked. Here, “cost” corresponds to a network resource, such as energy in wireless sensor networks (WSNs) or bandwidth in wired networks.

In the 3-Player Scenario, we show that communication is fundamentally cheaper than preventing communication. Specifically, we describe a protocol that guarantees correct transmission of m , and given that Carol incurs a cost of B , has the following properties. In Case 1, the expected cost to both Alice and Bob is $O(B^{\varphi-1} + 1)$ where φ is the golden ratio. In Case 2, the expected cost to both Alice and Bob is $O(B^{0.5} + 1)$. In both cases, Carol’s cost asymptotically exceeds the expected cost of either correct player. We extend these results and apply them to network applications in both the wireless and wired domains. Here too, we provide an analysis in terms of resource expenditures by the correct network devices relative to those of the adversarial devices.

1.1 Resource Competitiveness

In many situations, it is useful to view the performance of an algorithm \mathcal{A} in terms of its “competitiveness” and this notion underlies competitive analysis and approximation algorithms. In the case where \mathcal{A} is an online algorithm, a standard technique is to analyze the worst-case performance \mathcal{A} relative to an optimal algorithm \mathcal{OPT} that has full information regarding the input (ie. an optimal offline algorithm). For an optimization problem involving the minimization of an objective function (also referred to as a *cost function*) C , which takes as input an algorithm and an instance I from the set \mathcal{I} of all possible inputs, we say that \mathcal{A} achieves a competitive ratio of λ if $C(\mathcal{A}, I) \leq \lambda \cdot C(\mathcal{OPT}, I)$ for all $I \in \mathcal{I}$. This notion of *competitive analysis* was first introduced by Tarjan and Sleator [62] and it quantifies the amount by which \mathcal{A} is outperformed by \mathcal{OPT} . A closely related technique for algorithm analysis is that of establishing approximation guarantees. In this case, a similar definition applies with the major difference being that the input is fully known to \mathcal{A} *a priori*. Both the notions of a competitive ratio and an approximation ratio allow us to measure performance relative to an optimal algorithm under worst-case inputs which are often thought of as having been selected by an adversary.

A related metric is offered by game theory [51, 59]. Devices (or players) are assumed to be autonomous and governed by self-interested but rational behaviour. Here, the notion of the “price of anarchy” is the ratio of the worst-case Nash equilibrium to the global social optimum. For a system where each rational player acts to minimize their own respective cost function (or maximize their respective payoff function), the price of anarchy gauges the competitiveness of this system relative to the ideal scenario where all players act unselfishly to minimize global cost.

The approach taken in this paper provides a new notion of “competitiveness” for an n -player system that we call *resource competitiveness*. Each player p_i is classified as either *correct*, if its actions are prescribed by an algorithm \mathcal{A} , or *faulty* otherwise; let P be the set of correct players and F be the set of faulty players. The faulty players may collude and coordinate their attacks; therefore, we may view the players in F as being controlled by a single adversary. Let \mathcal{C}_i denote the resource expenditure (or cost) to a player p_i . If p_i is correct, then \mathcal{C}_i is the cost incurred by the actions prescribed by \mathcal{A} . Otherwise, p_i is faulty and \mathcal{C}_i is the cost incurred by pursuing an arbitrary strategy. The full memberships of P and F are not necessarily known to \mathcal{A} *a priori* and may be, in part, selected adversarially.

Given this notation, our most general formulation of resource competitiveness is as follows. We say that algorithm \mathcal{A} is ρ -resource-competitive if $\rho \cdot g(\{\mathcal{C}_i\}_{p_i \in P}) \leq a(\{\mathcal{C}_j\}_{p_j \in F})$ for any strategy employed

by the adversary and where $g(\cdot)$ and $a(\cdot)$ are functions that take as input the set of costs for the correct and faulty players, respectively. We call the parameter ρ the *resource-competitive ratio* and it measures the ratio $g(\cdot)$ to $a(\cdot)$. As a concrete example, in this paper, $g(\cdot)$ outputs the maximum resource expenditure by any correct player, and $a(\cdot)$ outputs the sum total resource expenditure by the faulty players $B = \sum_{p_j \in F} C_j$. Therefore, we may rewrite our definition as: $\rho \cdot \max_{p_i \in P} \{C_i\} \leq B$. In this case, ρ measures the ratio of the sum total resource expenditure by the adversary to the maximum resource expenditure by any correct player; a large ρ is desirable.

The generality of our definition allows us to capture the notion of “fairness” or “relative load-balancing”. In this work, our definition of $g(\cdot)$ motivates algorithms where it is important to minimize the worst-case cost to any single correct player. However, one can imagine cases where the sum total cost incurred by the correct players is the primary concern. Alternatively, it may be the case that the adversary is constrained such that the worst-case cost to any faulty player is a more suitable metric; for example, if the adversary wishes to maximize the lifetime of all faulty devices in a network where energy is scarce such as in a wireless sensor network.

1.2 Practical Aspects of Resource Competitiveness

As is often done with competitive ratios and approximation ratios, we will abuse the “ \leq ” notation in our definition of resource competitiveness by presenting ρ in terms of asymptotic notation when stating our results. In the case of an approximation ratio, this implies that the guarantee holds for sufficiently large problem instances. If approximation algorithms are being employed, they are typically used on optimization problems of large size and, therefore, the constant hidden within the asymptotic notation may not be a critical concern. In the case of resource competitiveness, the situation requires a little more consideration. Certainly, as B goes to infinity, all players enjoy an advantage over the adversary. However, due to the constants inside the asymptotic notation, it is possible that up to a threshold value for B the adversary is actually incurring less cost than some (or maybe all) of the correct players.

Given this situation, can the adversary simply launch “small” attacks? That is, what if the adversary attacks in a way that never exceeds the threshold value and, therefore, forces at least one correct player to spend more. If this occurs, it is critical to note that the amount of disruption caused by the adversary is also limited. In the context of the communication problems considered in this work, this implies that the duration for which the adversary may delay communication is limited. That is, *the adversary cannot both (i) force the correct players to spend more resources and (ii) prevent the correct players from communicating for long periods of time*. More generally, the adversary cannot force the correct players to incur greater cost while also preventing the successful termination of \mathcal{A} for a duration of time that depends on the constants involved. For our purposes, this means that the network throughput will be relatively high given that the constants inside the asymptotic notation are tolerable. We argue that, in the context of our applications, this is the case.

In the wireless setting, our cost is measured as the number of time slots for which the device is in an active state (ie. sending or listening). Notably, each slot is on the order of milliseconds. Even if the adversary never incurs a large resource expenditure in attempting to block communication, the battery on a wireless device will permit activity over large number of time slots and so permit the successful transmission of many messages before it expires. Therefore, the adversary can pursue a strategy of using small amounts of disruption in order to force a low value for ρ , but it will not achieve its aim of preventing communication. In the case of DDoS attacks in wired networks, where cost is measured in bandwidth, the actual bandwidth values in practice are on the order of megabits; therefore, this constant does not seem problematic. The adversary may opt to use far less bandwidth in its attacks; however, again, this does not result in a very effective DDoS attack.

This new metric of resource competitiveness differs from other prominent approaches to several cru-

cial ways. First, resource competitiveness measures the performance of an algorithm relative to the performance of an adversary rather than some global optimum. Specifically, unlike competitive analysis or approximation guarantees that are concerned with worst-case input instances (perhaps selected by an adversary), resource-competitiveness applies to situations where an adversary is truly present in the system rather than, for example, serving as an abstract representation of detrimental environmental effects. Second, while we still deal with players, each player either obeys a specified protocol (ie. is a correct player) or behaves in a faulty manner. In the latter case, this behaviour is assumed to be coordinated by an adversary and, in the worst case, we may consider faulty players to be pursuing arbitrary strategies; this is in contrast to the rational players assumed by game-theoretic approaches. Critically, we point out that for systems of n players, our definition places no bounds on the number of faulty players.

We feel that resource-competitive analysis is relevant to the area of fault-tolerant distributed computing where *all* players are resource-constrained. This notion incorporates the spirit of relative performance that underlies competitive analysis, approximation algorithms, and game theory. Furthermore, resource competitiveness captures an economic flavor while allowing for non-rational, and even malicious, behaviour by players. In this work, we obtain resource-competitive algorithms for fundamental scenarios regarding communication between a single sender and a single receiver, as well as between a single sender and multiple receivers. However, we believe that resource-competitive analysis will prove useful for analyzing a number of other distributed computing problems and we consider this in Section 5.

1.3 The 3-Player Scenario

We now describe the model parameters of the 3-Player Scenario.

Las Vegas Property: Communication of m from Alice to (a correct) Bob must be guaranteed with probability 1; that is, we require a Las Vegas protocol for solving the 3-Player Scenario. An obvious motivation for this Las Vegas property is a critical application, such as an early warning detection system or the dissemination of a crucial security update, where minimizing the probability of failure is paramount. The Las Vegas property has additional merit in multi-hop WSNs where Monte Carlo algorithms may not be able to achieve a sufficiently low probability of error; we expand on this in Section 3.6.3.

Channel Utilization: Sending or listening on the communication channel by Alice and Bob is measured in discrete units called *slots*. For example, in WSNs, a slot may correspond to an actual time slot in a time division multiple access (TDMA) type access control protocol. The cost for sending or listening is S or L per slot, respectively. When Carol blocks a slot, she disrupts the channel such that no communication is possible; blocking costs J per slot. If a slot contains traffic or is blocked, this is detectable by a player who is *listening* at the *receiving end* of the channel, but not by the originator of the transmission. For example, a transmission (blocked or otherwise) from Bob to Alice is detectable only by Alice; likewise, a transmission (blocked or otherwise) from Alice to Bob is detectable only by Bob. A player cannot discern whether a blocked slot has disrupted a legitimate message; only the disruption is detectable. For example, high-energy noise is detectable over the wireless channel in WSNs, but a receiving device cannot tell if this results from a message collision or a device deliberately disrupting the channel. We let B be the total amount Carol will spend over the course of the algorithm; this value is *unknown* to either Alice or Bob. Finally, we say that any player is *active* in a slot if that player is sending, listening or blocking in that slot.

Correct and Faulty Players: If Alice is faulty, there is clearly no hope of communicating m ; therefore, Alice is assumed to be correct. Regarding the correctness of Bob, in Case 1, Carol may spoof or control Bob; in Case 2, communications from Bob are always trustworthy. We emphasize that, in Case 1, Alice is uncertain about whether to trust Bob since he may be faulty. This uncertainty corresponds to scenarios where a trusted dealer attempts to disseminate content to its neighbors, some of whom may be spoofed or have suffered a Byzantine fault and used in an attempt to consume resources by requesting

numerous retransmissions. The more benign Case 2 corresponds to situations where communications sent by Bob are never disrupted and can be trusted; here, the blocking of m being sent from Alice to Bob is the only obstacle.

In the context of our definition of resource-competitiveness, in Case 1 of the 3-Player Scenario we have Alice as a member of P . Clearly, Carol belongs to set F . In the case of Bob, there is ambiguity since he may belong to P or he may belong to F . Indeed, Bob's membership is assumed to be controlled by Carol since she decides whether to spoof Bob or control his actions (placing him in F) or do neither (placing him in P). Our algorithm must achieve resource competitiveness while being oblivious to Bob's true membership. Lastly, in Case 2, it is clear that both Alice and Bob belong to P while Carol is the sole member of F .

Types of Adversary: Carol has full knowledge of past actions by Alice and Bob. This allows for *adaptive attacks* whereby Carol may alter her behaviour based on observations she has collected over time. Furthermore, under conditions discussed in Section 2.2, Carol can also be *reactive*: in any slot, she may detect a transmission and then disrupt the communication, however, we assume that she cannot detect when a player is listening. This is pertinent to WSNs where the effectiveness of a reactive adversary has been shown experimentally.

1.4 Solving the 3-Player Scenario: Resource-Competitive Algorithms

We analyze the cost of our algorithms as a function of B to obtain a notion of cost incurred by a player that is *relative to the cost incurred by Carol*. In devising our resource-competitive algorithms, we desire (but do not always perfectly achieve) two properties.

First, we clearly desire *favourable* protocols; that is, for B sufficiently large, Alice and Bob both incur asymptotically less expected cost than Carol. DoS attacks are effective because a correct device is *always* forced to incur a higher cost relative to an attacker. However, if the correct players incur asymptotically less cost than Carol, then Alice and Bob enjoy the advantage, and Carol is faced with the problem of having her resources consumed disproportionately in her attempt to prevent communication.

Second, our protocol should be *fair*; that is, Alice and Bob should incur the same *worst case asymptotic cost* relative to the adversary. When network devices have similar resource constraints, such as in WSNs where devices are typically battery powered, this is critical. Alternatively, in networks where a collection of resource-scarce devices (i.e. client machines represented by Alice) occupy one side of the communication channel and a single well-provisioned device (i.e. a server represented by Bob) occupies the other side, the *aggregate* cost to Alice's side should be roughly equal to that of Bob.

Recall our definition of resource competitiveness: $\rho \cdot \max_{p_i \in P} \{C_i\} \leq \sum_{p_j \in F} C_j$ for any strategies used by the faulty players and for $\sum_{p_j \in F} C_j > 0$. Favourability is clear from the definition and the degree of favourability corresponds to ρ . By considering the maximum cost to any of the correct players, the resource-competitive ratio is improved by spreading the cost equally over all players in \mathcal{F} ; therefore, fairness is also motivated. However, we will see results where *perfect* fairness is not achieved, but rather where the n players incur resource expenditures that are equal up to a polylogarithmic factor in terms of n , or where we consider situations where a player may be disproportionately well-provisioned and, therefore, absorb more cost in terms of resources. We expand on this when we examine applications in Section 3 and Section 4.

1.5 Our Main Contributions

Throughout, let $\varphi = (1 + \sqrt{5})/2$ denote the golden ratio. We also draw attention to the well-known relationship that $\Phi = \varphi - 1 = 1/\varphi \approx 0.62$ where Φ is known as the *golden ratio conjugate*. Our results

can be presented using Φ rather than φ ; however, for readability, we utilize φ throughout. We assume that S , L , and J are fixed constants. Our main analytical contributions are listed below.

Theorem 1.1. *Assume Carol is an adaptive adversary and that she is active for B slots. There exists a resource-competitive algorithm for the 3-Player Scenario with the following properties:*

- *In Case 1, the expected cost to each correct player is $O(B^{\varphi-1} + 1) = O(B^{0.62} + 1)$. In Case 2, the expected cost to each correct player is $O(B^{0.5} + 1)$.*
- *If Bob is correct, then transmission of m is guaranteed and each correct player terminates within $O(B^\varphi)$ slots in expectation.*

In other words, we show the existence of a $\Omega(B^{0.38})$ -resource-competitive algorithm for Case 1 and a $\Omega(B^{0.5})$ -resource-competitive algorithm for Case 2. In networks with sufficient traffic, Theorem 1.1 still holds when Carol is also reactive (Section 2.2). We also prove that any protocol which achieves $o(B^{0.5})$ expected cost for Bob requires more than $2B$ slots to terminate (Section 2.3); this lower bound has bearing on the worst-case $\omega(B)$ slots required by our protocol.

We next consider a more general setting where Alice wishes to locally (single-hop) broadcast to n neighboring receivers of which any number are spoofed or controlled by Carol. We emphasize that we place no bounds on the number of faulty receivers (indeed, all of them may be in F) and their respective membership in P or F is known only to Carol. Unfortunately, a naive solution of having each receiver execute a separate instance of our 3-Player Scenario protocol fails to be remotely fair. Thus, we need a different algorithm to achieve the following result.

Theorem 1.2. *There exists a resource-competitive algorithm for achieving local broadcast with the following properties:*

- *If Carol's receivers are active for a total of B slots, then the expected cost to Alice is $O(B^{\varphi-1} \ln n + \ln^\varphi n)$ and the expected cost to any correct receiver is $O(B^{\varphi-1} + \ln n)$.*
- *Transmission of m is guaranteed and all correct players terminate within $O((B + \ln^{\varphi-1} n)^{\varphi+1})$ slots (not in expectation). For $B \geq \ln^{\varphi-1} n$, this is within an $O(B^\varphi)$ -factor of the optimal latency.*

Therefore, Theorem 1.2 proves the existence of a $\tilde{\Omega}(B^{0.38})$ -resource-competitive algorithm for the case of single sender and n receivers; this algorithm is fair up to polylogarithmic factors. This result also holds for a reactive adversary, assuming a sufficient level of non-critical network traffic. Since the adversary may simply jam for the first B slots, it is impossible to achieve communication in less than $B + 1$ slots; it is this value to which we refer when speaking of the *optimal latency*.

Our last two theorems involve the application of our previous resource-competitive results to WSNs and wired networks, respectively. Reliable broadcast in *multi-hop* WSNs deals with conveying m from one node to all other nodes in the network. We make the standard assumptions that any node p can be heard by the set of neighboring nodes in the topology, $N(p)$ and that, for any p , at most t nodes in $N(p)$ suffer a fault [12, 13, 37]. We analyze the grid model using the result of Bhandari and Vaidya [13], and general graphs using the Certified Propagation Protocol (CPA) protocol of Pelc and Peleg [52].

Theorem 1.3. *For each correct node p , assume the t nodes in $N(p)$ are Byzantine and can be used by Carol to disrupt p 's communications for $\beta \leq B_0$ time slots where B_0 is a known positive value. Then, for β sufficiently large, using the local broadcast protocol of Theorem 1.2, each correct node incurs a cost of $o(\beta)$ while achieving reliable broadcast under the following topologies:*

- *In the grid with the optimal fault tolerance $t < (r/2)(2r + 1)$.*

- In any graph, assuming that (a) t is appropriately bounded such that CPA achieves reliable broadcast and (b) the topology and location of the dealer is known to all correct nodes.

To the best of our knowledge, all previous reliable broadcast protocols require each correct node to spend more energy in attempting communication than that spent in total by adversarial nodes. Specifically, in [15] where address spoofing and collision detection are considered, each correct node is required to be active for $B_0 + 1$ slots. Our results are the first to achieve favourability and, importantly, the first to account for the *significant cost of listening* to the wireless channel. While favourability is achieved only for $\beta = \omega(t \ln^{\varphi+1} t)$ (see Section 3.6.2 for the analysis), smaller values of β imply that the adversary cannot delay reliable broadcast for very long and that the cost of this interference to correct players is tolerable. This aligns with our discussion in Section 1.1 regarding the trade-off between throughput and ρ .

We note that the above result requires that the number of faulty nodes is bounded. This does not contradict our definition of resource competitiveness which places no bounds on the cardinality of F . Rather, this constraint on the relationship between the topology and the number of faulty nodes arises out of this *particular* application of our resource-competitive algorithms (and it is made by all previous works on the topic of reliable broadcast in this setting). That is, we distinguish between our resource-competitive results, used in our applications as primitives, and the applications of these primitives.

Finally, Theorem 1.4 is an application of Case 2 of Theorem 1.1 to the wired client-server scenario where Carol represents any number of malicious clients engaging in an application-level DDoS attack on a server.

Theorem 1.4. *Assume Carol commits her DDoS attack using a bandwidth R . Then, zero throughput is avoided if the expected aggregate bandwidth (upstream or downstream bits per second) of both the clients and the server is $G = O(R^{0.5})$, and the probability of a serviced request is $G/(G + R)$.*

Therefore, against a server defended by our protocol, Carol must incur additional monetary costs in order to procure the number of machines necessary for sustaining the level of attack that she would otherwise achieve. Also, we note that in this application, constraints on the cardinality of F are not necessary.

1.6 Related Work

Jamming Attacks in WSNs: Several works addressing applied security considerations demonstrate that devices in a WSN are vulnerable to adversarial jamming [3, 8, 41, 74] where the adversary deliberately disrupts the communication medium. Defenses include spread spectrum techniques (frequency or channel hopping), mapping with rerouting, and others (see [43, 49, 72, 73] and references therein). Recent applied work by Ashraf *et al.* [4] investigates a similar line of reasoning by examining ways in which multi-block payloads (each block in a packet has its own cyclic redundancy code), so-called “look-alike packets”, and randomized wakeup times for receivers can be used to force the adversary into expending more energy in order to jam transmissions. The authors’ approach is interesting and the use of look-alike packets to prevent an adversary from being able to differentiate between different types of network traffic is similar to our approach to foiling reactive jammers (see Section 2.2). However, on the whole, their approach is quite different from our own; moreover, it appears that the authors of [4] do not provide analytical results.

There are a number of theoretical results on jamming adversaries; however, none explicitly accounts for listening costs and there is no notion of favourability. Closely related work by Gilbert *et al.* [27] examine the duration for which communication between two players can be disrupted in a model with collision detection in a time-slotted network against an adversary who interferes with an unknown number of transmissions. As we do here, the authors assume channel traffic is always detectable at the receiving end (i.e. silence cannot be “forged”). The authors employ the notion of *jamming gain* which is, roughly speaking, the ratio of the duration of the disruption to the adversary’s cost for causing such disruption.

This is an interesting metric which can gauge the efficiency of the adversary’s attack; however, it does not incorporate the cost incurred by the correct players in the system. Pelc and Peleg [53] examine an adversary that randomly corrupts messages; we do not require the adversary to behave randomly. Awerbuch *et al.* [6] give a jamming-resistant MAC protocol in a single-hop network with an adaptive, but non-reactive, adversary. Richa *et al.* [58] significantly extend this work to multi-hop networks. Dolev *et al.* [21] address a variant of the gossiping problem when multiple channels are jammed. Gilbert *et al.* [26] derive bounds on the time required for information exchange when a reactive adversary jams multiple channels. Meier *et al.* [46] examine the delay introduced by a jamming adversary for the problem of node discovery, again in a multi-channel setting. Dolev *et al.* [22] address secure communication using multiple channels with a non-reactive adversary. Recently, Dolev *et al.* [20] consider wireless synchronization in the presence of a jamming adversary. We refer the reader to the survey by Young and Boutaba [76] for a more in-depth review of this area.

A distinguishing feature of our results is that in all of these previous works, the notion of cost relative to the adversary is virtually ignored. The work that comes closest to our approach is that of Gilbert *et al.* [27] in their consideration of jamming gain which is roughly defined as the ratio of the duration of the disruption to the adversary’s cost for causing the disruption. However, this is very different from resource competitiveness which involves a comparison of the correct players’ cost to that of the adversary. Another significant difference between our work and those previous is that our notion of cost incorporates both sending and listening costs, the latter of which has been largely overlooked. Surprisingly, listening costs are substantial. For example, the send and listen costs for the popular Telos motes [55] are 38mW and 35mW, respectively, and these are the dominant costs for an active device. A similar relationship between the send and listen costs holds for the well-known MICA family of devices [17, 18].

There have also been a number of game-theoretic treatments of wireless jamming. We refer the interested reader to the survey by Manshaei *et al.* [45] for a comprehensive treatment of this area. In comparing game-theoretic approaches to our work here, there is common ground in capturing the economics of a system involving multiple players. Another shared trait is that a relative “cost” or resource expenditure. However, as discussed in Section 1.1, a major differentiating feature of our model is that here we do not assume that all players are rational. Instead, we differentiate between those players that obey a prescribed strategy (dictated by a distributed algorithm) and those that may pursue an arbitrary strategy. The former model is well-suited to situations where the agents are completely autonomous and behave indifferently of each other, but not adversarially, to minimize their respective individual costs. The latter model addresses systems where agents are not completely autonomous, but are rather coordinated by a single administrative entity, and truly adversarial behaviour is present.

Reliable Broadcast: Reliable broadcast has been extensively studied in the grid model [10, 12–14, 34, 35, 37, 67]. Listening costs are accounted for by King *et al.* [34, 35] but jamming adversaries are not considered; however, the authors introduce the *Bad Santa* problem which we use to achieve a lower bound result in Section 2.3. With a reactive jamming adversary, Bhandhari *et al.* [15] give a reliable broadcast protocol when the amount of jamming is bounded and known *a priori*; however, correct nodes must expend considerably more energy than the adversary. Progress towards fewer broadcasts is made by Bertier *et al.* [11]; however, each node spends significant time in the costly listening state. Alistarh *et al.* [1] assume collision detection and achieve non-cryptographic authenticated reliable broadcast. They apply their result to the grid model with a reactive jamming adversary and also make use of the “unforgeability of silence” introduced in [27]; however, their algorithm, like previous algorithms for the grid, require that devices incur considerable listening costs.

Application-Level DDoS Attacks: At the application layer, DDoS attacks typically involve attackers masquerading as legitimate clients by sending a large volume of proper requests with the aim of overwhelming the computational resources of a server. This is in contrast to flooding attacks which achieve disruption by

depleting the bandwidth of the network; such attacks typically require significant bandwidth and attackers are more easily identified due to the out-of-band traffic. Application-level DoS attacks are common with recorded attacks on high-profile companies such as Yahoo, Amazon, CNN, eBay, and many others [25]. Proposals for dealing with DDoS attacks include over-provisioning [56], throttling techniques [28, 47], and currency schemes (see [5, 31, 69] and references therein). Here, we are focused on currency schemes, where the server provides service only to a client who pays in some form of currency. In [69], bandwidth is used as currency and, if the clients' aggregate bandwidth exceeds that of the attackers, then the clients capture server resources. Our work is complementary by delineating bounds on the expected bandwidth required to guarantee that the correct clients avoid zero throughput.

Lastly, a preliminary version of this work appeared in [36]. This current version contains many algorithms, results, and proofs that were previously omitted. Additionally, we have greatly expanded our exposition in several sections in order to provide a self-contained document and provide a fuller discussion of the ideas that were proposed in the original work.

2 The 3-Player Scenario Protocol

Figure 1 gives the pseudocode for our protocol called 3-PLAYER SCENARIO PROTOCOL (3PSP). Each round $i \geq 2$ consists of 2 phases and c is a constant to be determined later. We summarize a round i :

- *Send Phase*: This phase consists of 2^{ci} slots. In each slot: Alice sends m with probability $\frac{2}{2^i}$ for an expected total of $2^{(c-1)i+1}$ slots and Bob listens with probability $\frac{2}{2^{(c-1)i}}$ for an expected total of 2^{i+1} slots.
- *Nack Phase*: This phase consists of 2^i slots. If Bob has not received m , then Bob sends a request for retransmission, `req`, for all 2^i slots. This request is essentially a *negative acknowledgement* or a *nack*. Alice listens in each slot with probability $4/2^i$ (note that $i \geq 2$ is required) for an expected total 4 slots.

Termination Conditions: Termination conditions are important because Carol cannot be allowed to keep the players active in perpetuity while simultaneously forcing them to incur a *higher* cost. Bob terminates the protocol upon receiving m . Since Alice is not spoofed, as discussed in Section 1.3, this termination condition suffices. Alice terminates if she listens to a slot in the Nack Phase which is not blocked and does not contain `req` message; since blocked slots are detectable by Alice (who is on the receiving end of a `req` message) while listening (Section 1.3), this condition suffices. That is, Alice continues into the next round if and only if (1) Alice listens to zero slots or (2) all slots listened to by Alice in the Nack Phase contain a blocked slot or `req`. We highlight the two situations where this condition is met:

- *Send Failure*: Bob is correct and has not received m .
- *Nack Failure*: Bob is faulty and sends `reqs`, or Bob is correct and terminated and Carol either spoofs `reqs` or blocks slots in order to trick Alice into thinking a valid `req` was indeed sent and/or blocked.

Nack Failures and Cases 1 & 2: Note that an “acknowledgement” occurs via silence in at least one slot in the Nack Phase. We say a *Nack Failure* occurs when Carol blocks for all slots in the Nack Phase.

In Case 1, a Nack Failure corresponds to a critical attack that can be employed in Nack Phase after the delivery of m . Carol can avoid the listening costs in the Send Phase, and then drain Alice's energy by making it appear as if Bob repeatedly did not receive m and is requesting a retransmission in the Nack Phase. This attack affects Alice only. Note that if Bob is actually correct, the attack is only effective once m is received since, if a correct Bob has not received m , a `req` will be issued anyway and the attack accomplishes nothing.

In Case 2, no blocking occurs in the Nack Phase and, therefore, no Nack Failure can occur. In fact, in Case 2, the Nack Phase can be shortened to a single slot where Bob sends his `req` and Alice listens; however, this does not change our cost analysis and our current presentation is more general.

3-PLAYER SCENARIO PROTOCOL for round $i \geq 2$

Send Phase: For each of the 2^{ci} slots do

- Alice sends m with probability $2/2^i$.
- Bob listens with probability $2/2^{(c-1)i}$.

If Bob received the message, then Bob terminates.

Nack Phase: For each of the 2^i slots do

- Bob sends a `req` message.
- Alice listens with probability $4/2^i$.

If Alice listened to a slot in the Nack Phase where no `req` message or blocking was detected, she terminates.

Figure 1: Pseudocode for 3-PLAYER SCENARIO PROTOCOL.

2.1 Analysis of the 3-Party Scenario Protocol

For a given round, we say it is a *send-blocking* round if Carol blocks at least half of the slots in the Send Phase; otherwise, it is a *non-send-blocking* round. Similarly, a *nack-blocking* round is a round where Carol blocks or spoofs `req` messages from Bob in at least half the slots in the Nack Phase; otherwise, it is *non-nack-blocking*. Throughout, assume ceilings on the number of active slots of a player if it is not an integer.

Bounds on c : Clearly, $c > 1$ or Bob's listening probability in the Send Phase is nonsensical. For Case 1, note that if $c \geq 2$, then the expected cost to Alice is at least as much as the expected cost to a potentially faulty/spoofed Bob. If Bob happens to be faulty/spoofed, then the cost to him for a Nack Failure is less than the expected cost to Alice since a faulty/spoofed Bob will simply not listen in the Send Phase; as discussed above, we must avoid this since it admits a draining attack against Alice. Therefore, we have $1 < c < 2$. For Case 2, since Bob is guaranteed to be correct, the acceptable range is $1 < c \leq 2$.

Lemma 2.1. *Consider a non-send-blocking round of 3-PLAYER SCENARIO PROTOCOL. The probability that a correct Bob does not receive the message from Alice is less than e^{-2} .*

Proof. Let $s = 2^{ci}$ be the number of slots in the Send Phase. Let p_A be the probability that Alice sends in a particular slot. Let p_B be the probability that Bob listens in a particular slot. Let $X_j = 1$ if the message is not delivered from Alice to Bob in the j^{th} slot. Then $Pr[m \text{ is not delivered in the Send Phase}] = Pr[X_1 X_2 \cdots X_s = 1] = Pr[X_s = 1 \mid X_1 X_2 \cdots X_{s-1} = 1] \cdot \prod_{i=1}^{s-1} Pr[X_i = 1]$. Let $q_j = 1$ if Carol does not block in slot j ; otherwise, let $q_j = 0$. The value of q_j can be selected arbitrarily by Carol. Then $Pr[X_i = 1 \mid X_1 X_2 \cdots X_{i-1} = 1] = 1 - p_A p_B q_j$ and substituting for each conditional probability, we have $Pr[X_1 X_2 \cdots X_s = 1] = (1 - p_A p_B q_1) \cdots (1 - p_A p_B q_s) = \prod_{j=1}^s (1 - p_A p_B q_j) \leq e^{-p_A p_B \sum_{j=1}^s q_j} < e^{-2}$ since $p_A p_B \sum_{j=1}^s q_j > (2/2^i)(2/2^{(c-1)i})(s/2) = (2/2^i)(2/2^{(c-1)i})(2^{ci}/2) = 2$ since the round is not send-blocking and so Carol blocks less than $s/2$ slots. \square

Note that Lemma 2.1 handles adaptive (but not reactive) adversaries. A simple but critical feature of tolerating adaptive adversaries is that the probability that a player is active in one slot is independent from the probability that the player is active in another slot. Therefore, knowing that a player was active for

k slots in the past conveys no information about future activity. Believing otherwise is the trap of the well-known “Gambler’s Fallacy” [66]. For reactive adversaries, we need only modify Lemma 2.1 as we do later.

Lemma 2.2. *Assume that Bob is correct and there are no send-blocking rounds and no nack-blocking rounds. Then, the expected cost of each player is $O(S + L) = O(1)$.*

Proof. Using Lemma 2.1, the expected cost to Alice is at most $\sum_{i=2}^{\infty} e^{-2(i-2)} \cdot (2 \cdot 2^{(c-1)i} \cdot S + 4 \cdot L) \leq \sum_{i=2}^{\infty} (e^{5-i} \cdot S + e^{2-2i} \cdot 4i \cdot L) = (e^5 \cdot S \cdot \sum_{i=2}^{\infty} e^{-i}) + (e^2 \cdot 4 \cdot L \cdot \sum_{i=2}^{\infty} e^{-2i}) = O(S + L) = O(1)$. Similarly, the expected cost to Bob is at most $\sum_{i=2}^{\infty} e^{-2(i-2)} \cdot (2^{i+1} \cdot L + 2^i \cdot S) \leq \sum_{i=2}^{\infty} (e^{5-i} \cdot L + e^{4-i} \cdot S) = O(S + L) = O(1)$ since S and L are constants. \square

Now consider when attacks may occur in the Nack Phase:

Lemma 2.3. *Assume that Bob has received m by round i and that round i is non-nack-blocking. Then the probability that Alice retransmits m in round $i + 1$ is less than e^{-2} .*

Proof. Let $s = 2^i$ be the number of slots in the Nack Phase and let $p = 4/2^i$ be the probability that Alice listens in a slot. For slot j , define X_j such that $X_j = 1$ if Alice does not terminate. Then $\Pr[\text{Alice retransmits } m \text{ in round } i + 1] = \Pr[X_1 X_2 \cdots X_s = 1]$. Let $q_j = 1$ if Carol does not block in slot j ; otherwise, let $q_j = 0$. The q_j values are determined arbitrarily by Carol. Since Alice terminates if and only if she listens and does not detect any activity, then $\Pr[X_j = 1] = (1 - pq_j)$. Therefore, $\Pr[X_1 X_2 \cdots X_s = 1] \leq e^{-p \sum_{j=1}^s q_j} < e^{-2}$. \square

Lemma 2.4. *Assume there is at least one send-blocking round. Then, the expected cost to Alice is $O(B^{(c-1)/c} + B^{(c-1)})$ and the expected cost to a correct Bob is $O(B^{1/c})$.*

Proof. We consider Case 1 and Case 2 with regards to Bob, discussed in Section 1.3. Let $i \geq 2$ be the last round which is send-blocking. Let $j \geq i$ be the last round which is nack-blocking; if no such nack-blocking round exists, then assume $j = 0$. In Case 1, the total cost to Carol is $B = \Omega(2^{ci} \cdot J + 2^j \cdot J) = \Omega(2^{ci} + 2^j)$ since J is a constant. In Case 2, only send-blocking occurs and so $B = \Omega(2^{ci} \cdot J)$.

Alice: We first calculate the expected cost to Alice prior to successfully transmitting m . In round i , Carol blocks the channel for at least $2^{ci}/2$ slots. Using Lemma 2.1, the expected cost to Alice prior to m being delivered is $O(2^{(c-1)i} \cdot S + i \cdot L) + \sum_{k=1}^{\infty} e^{-2(k-1)} \cdot (2 \cdot 2^{(c-1)(i+k)} \cdot S + k \cdot L) = O(2^{(c-1)i} \cdot S) = O(2^{(c-1)i})$ by the bounds on c and given that S and L are constants; note, this is the total cost to Alice for Case 2.

Now, using Lemma 2.3, we calculate the expected cost to Alice after delivery; this addresses nack-blocking rounds possible only in Case 1. By assumption, the last nack-blocking round occurs in round j and therefore Alice’s expected cost is $O(2^{(c-1)j} \cdot S + j \cdot L) + \sum_{k=1}^{\infty} e^{-2(k-1)} \cdot (2 \cdot 2^{(c-1)(j+k)} \cdot S + k \cdot L) = O(2^{(c-1)j} \cdot S)$ by the bounds on c . Therefore, the total expected cost to Alice is $O(2^{(c-1)i} \cdot S + 2^{(c-1)j} \cdot S) = O(2^{(c-1)i} + 2^{(c-1)j})$. Since $B = \Omega(2^{ci} + 2^j)$, this cost as a function of B is $O(B^{(c-1)/c} + B^{(c-1)})$.

Bob: Finally, assume Bob is correct. Using Lemma 2.1, Bob’s expected cost prior to receiving m is $O(2^{i+1} \cdot L + 2^i \cdot S) + \sum_{k=1}^{\infty} e^{-2(k-1)} \cdot (2 \cdot 2^{i+k} \cdot L + 2^{i+k} \cdot S) = O(2^i \cdot L + 2^i \cdot S) = O(2^i)$ since S and L are constants. Thus, the expected cost for Bob as a function of B is $O(B^{1/c})$. \square

We now give the proof for Theorem 1.1 stated in Section 1.5:

Proof of Theorem 1.1: In Case 1, Lemma 2.4 tells us that the expected cost to Alice and Bob in terms of B is $O(B^{(c-1)/c} + B^{(c-1)})$ and $O(B^{1/c})$, respectively. Therefore, the exponents of interest which control

the cost to each player are $(c - 1)/c$, $c - 1$, and $1/c$. The value of c that should be chosen must minimize $\max\{(c - 1)/c, c - 1, 1/c\}$ since we are interested in fair protocols. Given that $1 < c < 2$, we have $1/c > (c - 1)/c$. Therefore, we solve for c in $c - 1 = 1/c$, this gives $c = (1 + \sqrt{5})/2$ which is the golden ratio. By Lemma 2.2 and the above argument, the expected cost to each player is $O(B^{\varphi-1} + 1)$. In Case 2, Lemma 2.4 tells us that Alice's expected cost in terms of B is $O(B^{(c-1)/c})$ the exponents of interest are simply $(c - 1)/c$ and $1/c$; minimizing them yields $c = 2$. Therefore, the cost to each player is $O(B^{1/2} + 1)$.

Finally, define latency to be the number of slots that occur prior to termination by both correct players. Consider how many non-send-blocking or non-nack-blocking rounds *either* player may endure before terminating successfully; let X denote the random variable for this number of rounds. Then, $E[X] \leq 1 \cdot (1 - e^{-2}) + 2 \cdot e^{-2}(1 - e^{-2}) + 3 \cdot e^{-4}(1 - e^{-2}) + \dots = \sum_{i=1}^{\infty} i e^{2(1-i)}(1 - e^{-2}) = (1 - e^{-2})e^2 \sum_{i=1}^{\infty} i(e^{-2})^i$ by Lemmas 2.1 or 2.3. Therefore, $E[X] \leq 1/(1 - e^{-2}) = O(1)$ which translates into $O(1)$ time slots consumed by non-send or non-nack-blocking rounds. Now consider the send- or nack-blocking rounds; note that Carol is limited to at most $\lg(2B) + O(1)$ such rounds which translates to $O(B^\varphi)$ time slots. Therefore, regardless of how Carol blocks, the expected number of time slots prior to successful termination is $O(B^\varphi)$. \square

2.2 Tolerating a Reactive Adversary

Consider a reactive adversary Carol who can detect channel activity without cost, and then block the channel; this ability is possible in WSNs (see Section 3.1). In our 3-Player Scenario, Carol can now detect that m is being sent in the Send Phase and block it without fail. To address this powerful adversary, we consider the case where critical data, m , and more often, non-critical data m' , is sent over the channel by other participants in addition to Alice and Bob. Carol can detect the traffic; however, she cannot discern whether it is m or m' without listening to a portion of the communication (such as packet header information).

In a slot where channel activity is detected, even if Carol listens for a portion of the message, she incurs a substantial cost. Therefore, the cost to Carol is proportional to the number of messages to which she listens. Importantly, in the presence of m' , Carol's ability to detect traffic for free is unhelpful since m' provides "camouflage" for m . Certainly Carol may block *all active slots* to prevent transmission of m ; however, this is no different than blocking *all slots* in our original 3-Player Scenario.

As an extreme example, assume that *all* slots in the Send Phase are used either by Alice to send m (as per our protocol) or Dave, whose transmissions of m' do not interest Carol, and the probability that a slot is used by Dave is higher. Then detecting channel activity does not help Carol decide on whether to block; all slots are used. Regardless of how she decides to act, Carol can do no better than picking slots independent of whether she detects channel activity. In other words, channel activity is no longer useful in informing Carol's decisions about whether to block.

But assuming *all* slots are active is problematic: (1) How is this guaranteed or coordinated? (2) Doesn't this much background traffic interfere with Bob's ability to receive from Alice? Instead, assume that other network traffic occurs such that Carol will always detect traffic on at least a constant fraction of slots in the Send Phase. Note that this does not help her block transmissions by Alice since she does not know the total amount of traffic that she will detect. Now, not all slots will necessarily be active. Upon detecting traffic, can Carol listen to a portion of the message to discover if it is m or m' and then decide on whether to block? Yes, but this is roughly as expensive as simply blocking outright. So again, detecting channel activity does not inform Carol's decisions. This is the idea behind our analysis.

This corresponds to situations where communication occurs steadily between many participants or via several distributed applications, and Carol wishes to target only a critical few. If m and m' are sent over the channel in the same slot, the two messages collide and Bob receives neither. Define a slot as *active* if

either m or m' is sent in that slot. For this result only, redefine a send-blocking round as one where Carol listens or blocks for at least a $1/3$ -fraction of the *active* slots; otherwise, it is a *non-send-blocking round*. We provide a result analagous to Lemma 2.1.

Lemma 2.5. *Let Carol be an adaptive and reactive adversary. Then, in a non-send-blocking round of the 3-PLAYER SCENARIO PROTOCOL, the probability that Bob does not receive m from Alice is at most e^{-2} .*

Proof. Let $x = 2^{ci}$ be the number of slots in the Send Phase. Consider the set of slots used by all participants other than Alice. We assume these participants pick their slots at random to send, so that for any slot the probability is $2/3$ that the slot is chosen by at least one of them. Since we assume these messages m' are sent independently at random, then Chernoff bounds imply that, with high probability (i.e., $1 - 1/x^{c'}$ for constants c', ϵ and sufficiently large x) the number of slots y during which m' is sent is greater than $(2x/3)(1 - \epsilon)$ where x is the total number of slots in a phase. In the same way, assume the number of slots in which Alice sends is at least $a = (1 - \delta)xp_A = (1 - \delta)2^{(c-1)i+1}$ with probability $1 - 1/x^{c''}$ for a constant δ, c'' and sufficiently large x . The number of active slots sent by Alice or other participants is clearly at least y .

By definition of a non-send-blocking round, Carol listens to or blocks less than $x/3$ (active) slots. As Carol has no information about the source of a message sent in an active slot until she listens to it, her choice is independent of the source of the message. Given a slot that Alice sends on, there is at least a $1 - (x/3)/y$ chance it will not be listened to or blocked by Carol. The probability that this slot will not be used by another participant is $1/3$ and the probability that Bob will listen to the slot is p_B . Hence the probability of a successful transmission from Alice to Bob on a slot which Alice sends on is at least $p = (1 - x/(3y))(1/3)p_B = (1 - 1/(2(1 - \epsilon)))(1/3)p_B \geq (1/3 - (1 + \delta)/6)p_B$ for sufficiently large x (that reduces the size of δ) when $y > (1 - \epsilon)(2x/3)$. Therefore, we can write $p \geq (1/12)p_B$. The probability that all messages that Alice sends fail to be delivered is at most $(1 - p_B/12)^a + 2/x^{c''}$ where the last term is the probability of the bad event that y or a is small and $c'' > 0$ is a constant. Redefine $p_B = 24/((1 - \delta)2^{(c-1)i})$ where the value 24 is there to simply off-set the additive $2/x^{c''}$ term. Note that this constant factor increase in the listening probability does not change our asymptotic results and our analysis in Section 2.1 proceeds almost identically. Therefore, we then have $(1 - p_B/12)^a + 2/x^{c''} \leq e^{-2}$. \square

The 3-PLAYER SCENARIO PROTOCOL can be modified so that the initial value of i is large enough to render the error arising from the use of Chernoff bounds sufficiently small; we omit these details. Also, the required level of channel traffic detected by Carol is flexible and different values can be accomodated if the players' probabilities for sending and listening are modified appropriately in the 3-PLAYER SCENARIO PROTOCOL; our results hold asymptotically. Finally, we emphasize that Lemma 2.3 does not require modification. Carol cannot decide to block only when Alice is listening since detecting when a node is listening is impossible. Alternately, Carol cannot silence a `req` through (reactive) blocking since this is still interpreted as a retransmission request. Using Lemma 2.5, Theorem 1.1 follows as before.

Finally, we note that the conclusion of our argument aligns with claims put forth in empirical results on reactive jamming in WSNs; that is, such behaviour does not necessarily result in a more energy-efficient attack because the adversary must still be listening to the channel for broadcasts prior to committing itself to their disruption [74].

2.3 On Latency and Lower Bounds

In Section 1.6, we mentioned the *Bad Santa* problem which is described as follows. A child is presented with K boxes, one after another. When presented with each box, the child must immediately decide whether or not to open it. If the child does not open a box, it can never be revisited. Half the boxes have presents in them, but the decision as to which boxes have presents is made by an adversarial Santa who

wants the child to open as many empty boxes as possible. The goal is for the child to obtain a present *with probability 1*, while opening the smallest expected number of boxes. In [34, 35], the authors prove a lower bound of $\Omega(K^{0.5})$ on the expected number of opened boxes.

Theorem 2.6. *Any algorithm that solves the 3-Player Scenario with $o(B^{0.5})$ cost to Bob must have a latency exceeding $2B$.*

Proof. A lower bound for the 3-Player Scenario is complicated by the possibility that the strategies of Alice and Bob may adapt over time; for example, they may change depending on how Carol blocks. To address this, we assume a more powerful Bob. Specifically, assume that communication of m occurs if Bob is able to find an unblocked time slot in which to listen *or* to send. Furthermore, assume Bob can tell when he has found such a slot once he listens or sends in that slot. Therefore, such a Bob is at least as powerful as the Bob in the 3-Player Scenario.

Now, if Carol has a budget of size B , we ask: Does Bob have a strategy with $o(B^{0.5})$ expected active slots such that, with probability 1, he finds at least one unblocked slot within $2B$ slots? Assume that such a strategy exists and consider the Bad Santa problem on $2B$ boxes. Using Bob's strategy, the child is guaranteed to obtain a present with probability 1 while opening $o(B^{0.5})$ boxes in expectation. However, this contradicts the $\Omega(B^{0.5})$ lower bound result in [34] and the result follows. \square

This result illustrates a relationship between the Bad Santa problem and the 3-Player Scenario, and it provides some insight into why our protocol has a worst-case latency of $\omega(B)$ slots.

3 Application 1: Jamming Resistance in Wireless Sensor Networks

The shared wireless medium of sensor networks renders them vulnerable to jamming attacks [70]. A jamming attack occurs when an attacker transmits noise at high energy, possibly concurrently with a (legitimate) transmission, such that communication is disrupted within the area of interference. Consequently, this behaviour threatens the availability of sensor networks [72].

3.1 Rationale for the 3-Player Scenario Involving WSN Devices

Wireless network cards offer states such as *sleep*, *receive (or listen)* and *transmit (or send)*. While the sleep state requires negligible power, the cost of the send and listen states are roughly equivalent and dominate the operating cost of a device. For example, the send and listen costs for the popular Telos motes are 38mW and 35mW, respectively (note $S \approx L$) and the sleep state cost is $15\mu\text{W}$ [55]; therefore, the cost of the send/listen state is more than a factor of 2000 greater and the sleep state cost is negligible. Disruption may not require jamming an entire slot so we set $J < S$ and assume a small m such that J and S are within a constant factor of each other; larger messages can be sent piecewise. In our protocols, we account for both send and receive costs. Throughout, when a node is not active, we assume it is in the energy-efficient sleep state.

Slots: There is a single channel and a time division multiple access (TDMA)-like medium access control (MAC) protocol; that is, a time-slotted network. For example, the well-known LEACH [30] protocol is TDMA-based. For simplicity, a *global* broadcast schedule is assumed; however, this is likely avoidable if nodes maintain multiple schedules as with S-MAC [75]. Even then, global scheduling has been demonstrated by experimental work in [40] and secure synchronization has been shown [24].

A blocked slot occurs when Carol jams. Clear channel assessment (CCA), which subsumes carrier sensing, is a common feature on devices for detecting such events [57] and practical under the IEEE 802.11 standard [19]. *Collisions are only detectable by the receiver* [72]. When a collision occurs, a correct

node discards any received data. We assume that the absence of channel activity cannot be forged by the adversary; this aligns with the empirical work by Niculescu [50] who shows that channel interference increases linearly with the combined rate of the sources. Finally, we also note that several theoretical models feature collision detection (see [1, 6, 15, 27, 58]).

On Reactive Adversaries: CCA is performed via the radio chip using the *received signal strength indicator* (RSSI) [65]. If the RSSI value is below a clear channel threshold, then the channel is assumed to be clear [7]. Such detection consumes on the order of 10^{-6} W which is three orders of magnitude smaller than the send/listen costs; therefore, Carol can detect activity (but not message content) at essentially zero cost. Listening to even a small portion of a message costs on the order of milliwatts and our argument from Section 2.2 now applies.

Cryptographic Authentication: We assume that messages can be authenticated. Therefore, Carol cannot spoof Alice; however, Bob's `req` can essentially be spoofed by a Nack-Failure (as discussed in Section 2) which, along with jamming, makes the problem non-trivial. Several results show how light-weight cryptographic authentication can be implemented in sensor networks [32, 38, 42, 70, 71]; therefore, it is important to consider its impact as we do here. However, the adversary may capture a limited number of players (such as Bob); these players are said to suffer a Byzantine fault and are controlled by the adversary [70, 72]. Given this attack, we emphasize that, while we assume a shared key to achieve authentication, *attempts to share a secret send/listen schedule between Alice and Bob allows Carol to manipulate players in ways that are problematic*. This is discussed further in Section 3.3.

3.2 Local Broadcast and Guaranteed Latency

Our protocol LOCAL BROADCAST handles the general single-hop broadcast situation where Alice sends m to a set of n neighboring receivers within her transmission range. At first glance, this seems achievable by having each receiver execute an instance of 3PSP with Alice. However, the expected active time for Alice is an $\Omega(n)$ -factor larger than any correct receiver; thus, this is unfair. Furthermore, this protocol has poor latency. Here, we give a fast protocol that is both fair and favourable up to small polylogarithmic factors.

Our pseudocode is given in Figure 2 which is valid for $n \geq 2$. The probabilities for sending and listening are modified and there are two more phases (the Deterministic Send and Deterministic Nack Phases) where players act deterministically. Note that `req` messages can collide in the Probabilistic Nack Phase and will certainly collide in the Deterministic Nack Phase. This is correct as such a collision is due to either jamming or multiple receivers (correct or faulty) requesting a retransmission; this is fine and Alice will resend. LOCAL BROADCAST takes in as arguments the message m , the sender (Alice) and the set of receivers R_{Alice} . If the adversary jams, then none of the correct receivers receive m in that slot. We now prove the properties of LOCAL BROADCAST; for simplicity, we discard with the notation S , L , and J .

Lemma 3.1. *Consider a non-send-blocking round. The probability that at least one correct receiver does not receive the message from Alice is less than $1/n^2$.*

Proof. Let s be the number of slots in the Probabilistic Send Phase of round i . Let $p_A = 3 \ln n / 2^i$ be the probability that Alice transmits in a particular slot. Let $p_b = 2 / 2^{(\varphi-1)i}$ be the probability that a particular correct receiver b listens in a particular slot. Let $X_j = 1$ if the message is not transmitted from Alice to receiver b in the j^{th} slot. Then $\Pr[m \text{ is not successfully transmitted to the } b \text{ during the Probabilistic Send Phase}] = \Pr[X_1 X_2 \cdots X_s = 1] = \Pr[X_s = 1 \mid X_1 X_2 \cdots X_{s-1} = 1] \cdot \Pr[X_1 X_2 \cdots X_{s-1} = 1]$. Let $q_j = 1$ if the adversary does not jam given $X_1 X_2 \cdots X_{i-1}$; otherwise, let $q_j = 0$. The value of q_j can be selected arbitrarily by the adversary. Then $\Pr[X_i = 1 \mid X_1 \cdots X_{i-1} = 1] = 1 - p_A p_b q_i = 1 - (6 \ln n / 2^{\varphi i}) q_j$. Then

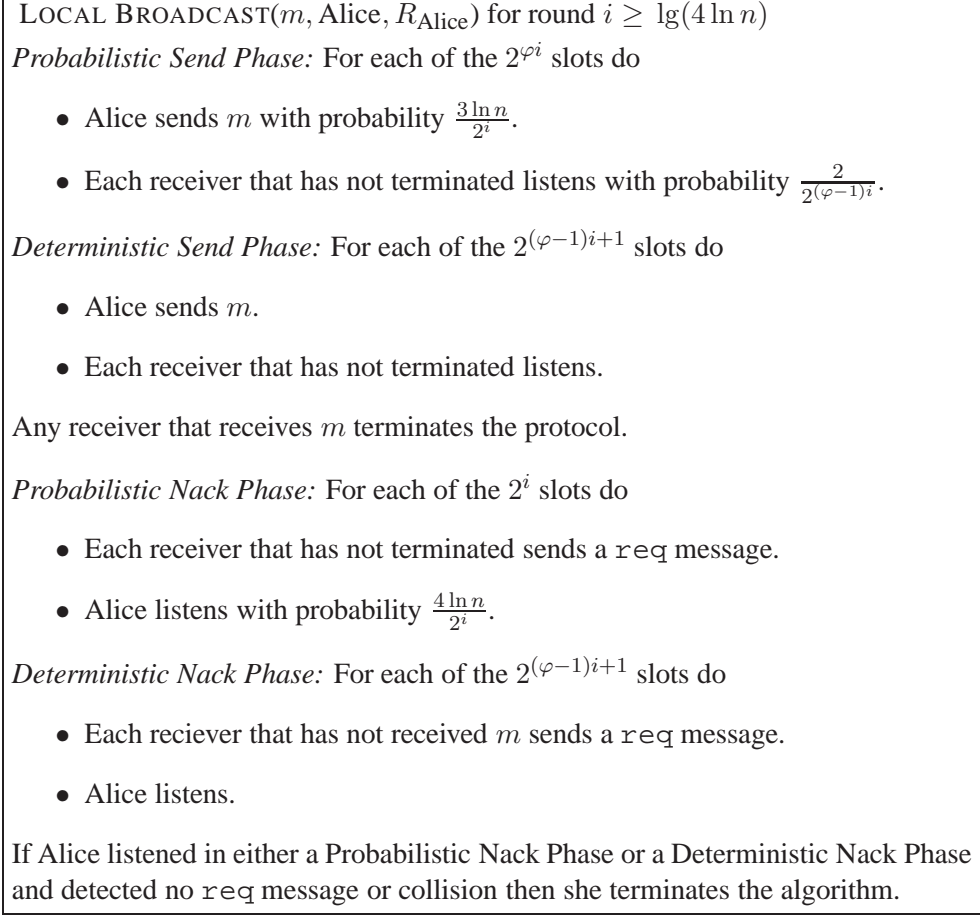


Figure 2: Pseudocode for LOCAL BROADCAST.

we have $\Pr[X_1 X_2 \cdots X_s = 1] = (1 - p_{Apb} q_1) \cdots (1 - p_{Apb} q_s) \leq \prod_{j=1}^s (1 - p_{Apb} q_j) \leq e^{-p_{Apb} \sum_{j=1}^s q_j} < 1/n^3$ since $p_{Apb} \sum q_j > (6 \ln n / 2^{\varphi^i}) \cdot (2^{\varphi^i} / 2) = 3 \ln n$ given that this is a non-send-blocking round. Taking a union bound, the probability that at least one correct receiver has not received m is less than n^{-2} . \square

A notable difference between Lemma 3.1 and the previous Lemma 2.1 is that here we want to bound the probability that *any* correct node has not received m . This requires a union bound which necessitates the $O(\log n)$ factor in our new analysis. A slightly tighter analysis may be possible; however, it should not change things asymptotically. We also note Lemma 3.1 can be modified to handle a reactive adversary in the same way as done for 3-PLAYER SCENARIO PROTOCOL; we omit the details.

Lemma 3.2. *Assume that by round i all correct receivers have heard the message m . Assume that round i is non-nack-blocking. Then the probability that Alice retransmits the message in round $i + 1$ is less than $1/n^2$.*

Proof. This is computed similarly to the proof of Lemma 3.1. Let s be the number of slots in the Probabilistic Nack Phase and let $p = 4/2^i$ be the probability that Alice listens in a slot. For slot j , define X_j such that $X_j = 1$ if Alice does not terminate. Then $\Pr[\text{Alice retransmits } m \text{ in round } i + 1] = \Pr[X_1 X_2 \cdots X_s = 1]$. Let $q_j = 1$ if the adversary does not jam given $X_1 X_2 \cdots X_{i-1}$; otherwise, let $q_j = 0$. The q_j values are determined arbitrarily by the adversary who controls the faulty receivers. Since

Alice terminates if and only if it listens and does not detect any activity, then $Pr[X_j = 1] = (1 - pq_j)$. Therefore, $Pr[X_1 X_2 \cdots X_s = 1] \leq e^{-p \sum_{j=1}^s q_j} < n^{-2}$ since $p \sum_{j=1}^s q_j > (4 \ln n / 2^i)(2^i / 2) = 2 \ln n$ given that this is a non-nack-blocking round. \square

Lemma 3.3. *Assume all receivers are correct and there are no send-blocking or nack-blocking rounds. Then the expected cost to Alice is $O(\ln^\varphi n)$ and the expected cost to any correct receiver is $O(\ln n)$.*

Proof. Let $d = \lg(4 \ln n)$ and $n \geq 3$. Using Lemma 3.1, the expected cost to Alice is at most:

$$\begin{aligned} & \sum_{i=d}^{\infty} n^{-2(i-d)} \cdot (2^{(\varphi-1)i} \cdot 3 \ln n + 2^{(\varphi-1)i+1} + 4 + 2^{(\varphi-1)i+1}) \\ &= O(\ln^\varphi n) + O\left(\ln n \cdot \sum_{k=1}^{\infty} \left(\frac{2^{(\varphi-1)}}{n^2}\right)^k\right) \\ &= O(\ln^\varphi n) \text{ by the geometric series.} \end{aligned}$$

Similarly, using Lemma 3.2, the expected cost to each receiver is at most:

$$\begin{aligned} & \sum_{i=d}^{\infty} n^{-2(i-d)} \cdot (2^{i+1} + 2^{(\varphi-1)i+1} + 2^i + 2^{(\varphi-1)i+1}) \\ &= O(\ln n) + O\left(\sum_{k=1}^{\infty} \left(\frac{2}{n^2}\right)^k\right) \\ &= O(\ln n) \text{ by the geometric series.} \end{aligned}$$

\square

Lemma 3.4. *Assume there is at least one send-blocking round. The expected cost to Alice is $O(B^{\varphi-1} \ln n + \ln^\varphi n)$ and the expected cost to any correct receiver is $O(B^{\varphi-1} + \ln n)$.*

Proof. Let $i \geq \lceil \lg(4 \ln n) \rceil$ be the last round which is send-blocking and let j be the last round which is nack-blocking, $j \geq i$; if no such nack-blocking round exists, then $j = 0$. Then the cost to the adversary is $B = O(2^{\varphi i} + 2^j)$.

Alice: Using Lemma 3.2, the expected cost to Alice prior to successfully terminating is $O(2^{(\varphi-1)i} \ln n) + \sum_{k=1}^{\infty} n^{-2(k-1)} \cdot O(2^{(\varphi-1)(j+k)} \ln n) = O(2^{(\varphi-1)i} \ln n + 2^{(\varphi-1)j} \ln n)$. Therefore, in terms of B , the cost to Alice is $O(B^{\varphi-1} \ln n)$ and by Lemma 3.3, Alice's total expected cost is $O(B^{\varphi-1} \ln n + \ln^\varphi n)$.

Correct Receivers: In the worst case, all rounds up to i have been send-blocking, in which case the expected cost to each correct receiver up to the end of round $i + 1$ is $O(2^i)$. Therefore, in terms of B , and using Lemma 3.3, the cost to each correct receiver is $O(B^{\varphi-1} + \ln n)$ noting that $1/\varphi = \varphi - 1$. \square

Lemma 3.5. *Alice and all correct receivers terminate LOCAL BROADCAST in $25 \cdot (B + 2^{\varphi-1} \ln^{\varphi-1} n)^{\varphi+1}$ time slots.*

Proof. The deterministic phases play a key role in establishing the bound on latency. If the adversary is not active for all slots in the deterministic Send Phase, then all correct receivers obtain m . Once all correct receivers terminate, the adversary must be active in all slots of the deterministic Nack Phase in order to prevent Alice from terminating. Therefore, prior to successful termination of all correct players (including Alice), the adversary is active for at least $2^{(\varphi-1)i+1}$ slots per round i in Epochs 2 and 4. For $d = \lg(4 \ln n)$, we seek the number of rounds ρ such that $\sum_{i=d}^{\rho} 2^{(\varphi-1)i+1} \geq B$ which yields that $\rho \geq \varphi \lg(B + 2^{\varphi-1} \ln^{\varphi-1} n)$ rounds suffices to exhaust the adversary (we are not being exact). Each round i has at most $4 \cdot 2^{\varphi i+1}$ slots so ρ rounds equal at most $25 \cdot (B + 2^{\varphi-1} \ln^{\varphi-1} n)^{\varphi+1}$ slots. \square

The value n is the number of devices within the broadcast range of Alice. Throughout, we have assumed that n is known *a priori*. There has been work done on node discovery in the presence of jamming (see Meier *et al.* [46]) and we assume that similar techniques can be used to obtain the value n . For a determined adversary, we expect $B > n$; that is, for an adversary intent on preventing communication, the number of time slots jammed will likely exceed the number of neighbors. Therefore, $B \gg \ln^{\varphi+1} n$. In this case (actually for $B \geq \ln^{\varphi-1} n$), the latency is $O(B^{\varphi+1})$ and, noting that Carol can prevent transmission for at least B slots, this is within an $O(B^{\varphi})$ -factor of the optimal latency. By this and Lemmas 3.3, 3.4, and 3.5, Theorem 1.2 now follows.

3.3 Why a Shared Schedule is Problematic

In our WSN application, we assume that messages from Alice can be authenticated using light-weight cryptographic techniques. Given this, we consider: might Alice and Bob (or even more players) also share a secret schedule? This would reduce the costs in Theorem 1.1 due to the Send Phase where neither player knows if the other is active with any certainty.

Unfortunately, such a schedule becomes known to the adversary if a player suffers a Byzantine fault and this causes problems in more general scenarios. For instance, consider the simple extension of Alice and two receivers. In our local broadcast problem, which is a key subroutine for our reliable broadcast protocol, Alice broadcasts to its two neighboring receivers concurrently in order to be fair. Therefore, both receivers must know when Alice transmits in the Send Phase. By corrupting one receiver, this schedule becomes known to the adversary who can then block transmissions by Alice perfectly and easily prevent the other receiver from receiving m . Clearly, this attack extends to the case where there are n receivers and Alice wants to achieve a local broadcast.

Other problems arise in a multi-hop scenario. For example, in our reliable broadcast protocol, each node listens to many different senders. A faulty receiver can interfere with many more senders by acting in the same manner as above for each of these senders. Therefore, by purposely avoiding a pre-set shared schedule, our use of randomness allows us to foil such attempts by the adversary.

3.4 Jamming-Resistant Reliable Broadcast: Obtaining Favourability

Reliable broadcast has been extensively studied in the multi-hop grid model [12–14, 35, 37], particularly with a jamming adversary [1, 11, 15]. Reliable broadcast is possible when t Byzantine nodes can each jam at most n_c transmissions [15]. Unfortunately, the protocol of [15], and the improvement by [11], requires that *correct nodes possess more energy than the Byzantine nodes*. In particular, while the sending costs are improved in [11], both [11, 15] allow the adversary to force a correct node to *listen* for $t \cdot n_c + 1$ slots where n_c is the number of times each node can cause a message collision (listening costs in [1] are similar). Therefore, while the total sum cost to the adversary is $t \cdot n_c$, each correct node may suffer a cost of $t \cdot n_c + 1$. This cost ratio affords the adversary a DDoS attack since these previous protocols are consistently unfavourable.

3.5 A Review of Reliable Broadcast in the Grid Model

For the purposes of being self contained, we begin with a review of the popular grid model of WSNs [10, 12–14, 34, 35, 37, 67]. Here, each point in a two-dimensional grid holds a node and we let $p(x, y)$ denote the node p at location (x, y) . Whenever a node sends a message, all listening nodes within L_∞ distance r can hear this message assuming there is no interference.¹ The notation $N(p)$ or $N(x, y)$ denotes the set

¹In the L_∞ metric, the distance between two points (x_1, y_1) and (x_2, y_2) is $\max\{|x_1 - x_2|, |y_1 - y_2|\}$.

RELIABLE BROADCAST PROTOCOL (BHANDARI AND VAIDYA, 2005)

- Initially, the source s does a local broadcast of message m .
- Each node $i \in N(s)$ commits to the first value it receives from s and does a one-time broadcast of $\text{COMMIT}(i, m)$.
- The following protocol is executed by each node j (including those nodes in the previous two steps):
 - On receipt of a $\text{COMMIT}(i, m)$ message from a neighbor i , node j records the message and broadcasts $\text{HEARD}(j, i, m)$.
 - On receipt of a $\text{HEARD}(j', i, m)$, node j records this message.
 - Upon receiving COMMIT or HEARD messages that 1) claim v as the correct value and 2) are received along at least $t + 1$ node disjoint paths that all lie within a single neighbourhood, then node j commits to v and does a one-time broadcast of $\text{COMMIT}(j, m)$.

Figure 3: The reliable broadcast protocol proposed by Bhandari and Vaidya for tolerating Byzantine faults in the grid model.

of nodes within radius r of a node $p(x, y)$; this is referred to as p 's *broadcast neighbourhood*. Within any broadcast neighbourhood, at most t nodes may suffer a fault; this is the “ t -locally bounded fault model”.

There is a dealer (also referred to as the *source* in the literature) located at $(0, 0)$ and all nodes are aware of this. There is also a global broadcast schedule that assigns each node p a time slot in which p may broadcast. In [37], an example broadcast schedule is provided: each node in position (x, y) broadcasts in time slot $((x \bmod (2r + 1)) \times (2r + 1) + (y \bmod (2r + 1))) \bmod 2r + 1)^2$. Finally, for simplicity, we assume that messages fit within a single slot

The results by Koo [37] and by Bhandari and Vaidya [13] prove that when faults are Byzantine, reliable broadcast is possible in the grid if and only if $t < (r/2)(2r + 1)$. Figure 3 provides the pseudocode for the reliable broadcast protocol of Bhandari and Vaidya [13] that achieves this optimal fault tolerance.

Proving that this protocol is correct is non-trivial. Here, for the purposes of being self contained, we provide a summary of the proof and refer the interested reader to [13] for a complete analysis. A review is also helpful in providing the groundwork for our following arguments which follow the same general structure. Throughout the protocol, the message $\text{COMMIT}(i, m)$ means that node i has committed to a message m , and the message $\text{HEARD}(j, i, m)$ means that node j has heard a message $\text{COMMIT}(i, m)$. We also require the notion of a perturbed neighbourhood $PN(p)$ of $p(a, b)$ as $PN(p) = N(a + 1, b) \cup N(a - 1, b) \cup N(a, b + 1) \cup N(a, b - 1)$. The proof in [13] proceeds by showing that for each node p in $PN(a, b) - N(a, b)$, there exist $2t + 1$ paths P_1, \dots, P_{2t+1} lying within a single neighbourhood $N(a, b + r + 1)$, each having one of the forms listed below:

- $P_i = (q, p)$ which is a one hop path $q \rightarrow p$ or
- $P_i = (q, q', p)$ which is a two hop path $q \rightarrow q' \rightarrow p$

where q, q', p are distinct nodes and q, q' belong to a single neighbourhood $N(a, b + r + 1)$, and $q \in N(a, b)$ where, crucially, the correct nodes in $N(a, b)$ have committed to the correct message. The existence of these $2t + 1$ paths, and the fact that each broadcast neighbourhood has at most $t < (r/2)(2r + 1)$ Byzantine faults, is sufficient to prove that reliable broadcast is achieved by the protocol. For simplicity, we can

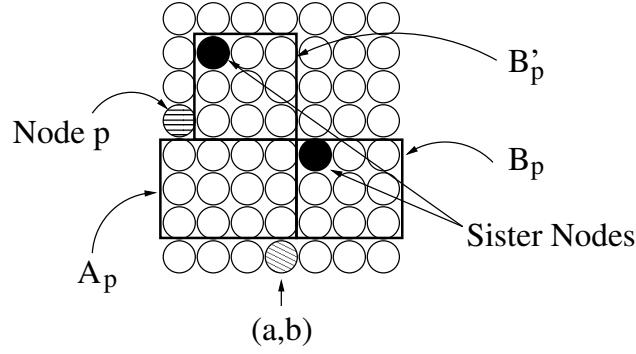


Figure 4: Depiction of the sets A_p , B_p , B'_p and sister nodes for a particular node p in the grid. Here $a, b, z = 0$ and $r = 3$ so the corridor C has width $2r + 1 = 7$.

consider $p \in N(a, b + 1)$ since the analysis is nearly identical for the cases where $p \in N(a + 1, b)$, $p \in N(a - 1, b)$, and $p \in N(a, b - 1)$.

The node p lies in $N(a, b + 1) - N(a, b)$ and can be considered to have location $(a - r + z, b + r + 1)$ where $0 \leq z \leq 2r$. Now, summarizing the proof in [13], we prove the existence of $r(2r + 1)$ node-disjoint paths $P_1, \dots, P_{r(2r+1)}$ all belonging to the same neighbourhood:

- **One-Hop Paths:** the set of nodes $A_p = \{q(x, y) \mid (a - r) \leq x \leq (a + z) \text{ and } (b + 1) \leq y \leq (b + r)\}$ belong to $N(a, b)$ and are neighbors of p . Therefore, there exist $r(r + z + 1)$ paths of the form $q \rightarrow p$ where $q \in A_p$.
- **Two-Hop Paths:** consider the sets $B_p = \{q(x, y) \mid (a + z + 1) \leq x \leq (a + r) \text{ and } (b + 1) \leq y \leq (b + r)\}$ and $B'_p = \{q'(x', y') \mid (a + z + 1 - r) \leq x' \leq a \text{ and } (b + r + 1) \leq y' \leq (b + 2r)\}$. The nodes in B_p are in $N(a, b)$ while the nodes in B'_p are in $N(p)$. Moreover, the set B'_p can be obtained by shifting left by r units and up by r units. Therefore, there exists a one-to-one mapping between the nodes in B_p and the nodes in B'_p . For $u \in B_p$, we call the corresponding node $u' \in B'_p$, the *sister node* of u . Note that each node has at most two sister nodes, and this can be seen in Figure 4. Therefore, there are $r(r - z)$ paths of the form $q \rightarrow q' \rightarrow p$.

In total, there are $r(r + z + 1) + r(r - z) = r(2r + 1)$ disjoint paths all belonging to the neighbourhood $N(a, b + r + 1)$. Figure 4 depicts aspects of the discussion above where $a, b = 0$. Now, note that the predecessor set $G_p = A_p \cup B'_p$ is the set of nodes to which p must listen in order to gather the information that allows p to commit to the correct message.

In this section, we extend our results for single-hop communication problems to achieve jamming-resistant protocols for reliable broadcast. Our protocols rely heavily on the results of Bhandari and Vaidya [13] discussed above. In particular, we assume that each node p knows a predecessor set G_p of nodes to which node p should listen for messages. As we have just seen, the existence of G_p is shown by the constructive proofs in [13]. Our protocols detail when each node p should listen to nodes in G_p and when each node p should broadcast its commitment message.

3.5.1 Our Reliable Broadcast Protocol in the Grid

As discussed, there are $t < (r/2)(2r+1)$ Byzantine nodes in any neighborhood. We reiterate our statement from Section 1.5 that this bound does not contradict our definition of resource competitiveness. Rather, as we have seen in the review above, this bound is necessary for otherwise reliable broadcast in the grid

is infeasible; this is a constraint imposed by the application considered here. Similarly, unlike our single-hop case, the amount of jamming in a neighborhood is upper bounded by B_0 and known. This is required in [11, 15] and a similar assumption is made in [6, 58]. B_0 represents the number of times a Byzantine node can deviate from the global schedule within some time frame in a neighborhood before being identified and subjected to defensive techniques (see [72]). Not exceeding B_0 in each time frame allows the adversary to attack throughout the lifetime of the network and we pessimistically assume that B_0 is large so that the adversary may inflict sustained attacks.

As with previous works, we assume that there is a global broadcast schedule that assigns each node a slot for broadcasting and that this schedule is obeyed by all correct nodes; the ordering is always the same but the actual specification is unimportant (see [37] for an example). A *cycle* is defined as on full pass through a global broadcast schedule—we call these *transmit slots*—plus an additional n slots that we call *response slots*; we expand on this later.

Overview of the Protocol: The pseudocode is in Figure 5. Our protocol synchronizes the timing of nodes for sending and listening. While this synchronization is not mathematically challenging, a full description yields an unreadable protocol. *For ease of exposition, our treatment addresses each node q in $C = \{q(x, y) \mid -r \leq x \leq r \wedge y \geq 0\}$; that is, a corridor of width $2r + 1$ moving up from d .* Traversing the x -coordinates is nearly identical and the grid can be covered piecewise by these two types of corridors.

For each node p , define $A_p = \{q(u, v) \mid (a - r) \leq u \leq (a + z) \text{ and } (b + 1) \leq v \leq (b + r)\}$, $B_p = \{q(u, v) \mid (a + z + 1) \leq u \leq (a + r) \text{ and } (b + 1) \leq v \leq (b + r)\}$ and $B'_p = \{q'(u', v') \mid (a + z + 1 - r) \leq u' \leq (a) \text{ and } (b + r + 1) \leq v' \leq (b + 2r)\}$ for $0 \leq z \leq r$. The set B'_p is obtained from B_p by shifting left by r units and up by r units; under this 1-to-1 translation, $q_1 \in B_p$ and $q_2 \in B'_p$ are sister nodes. The reader is referred to [13] or [35] for a more in-depth discussion of these sets.

While a full presentation is somewhat tedious, the main idea is that where a node would have broadcasted a message to a group of nodes in the protocol of Bhandari and Vaidya [13], we now use LOCAL BROADCAST to communicate a message to that group of nodes. In terms of the messages themselves, node q issues a COMMIT(q, m) message if q has committed to m . Node q_2 sends HEARD(q_2, q_1, m) if q_2 has received a message COMMIT(q_1, m). As in [13], p commits to m when it receives $t + 1$ COMMIT(q, m) or HEARD(q_2, q_1, m) from node-disjoint paths all lying within a single $(2r + 1) \times (2r + 1)$ area.

We now discuss how to move from slots in LOCAL BROADCAST to slots in a cycle. In general, the transmit slots in a cycle are used by a node p to transmit a HEARD or COMMIT message via LOCAL BROADCAST to a receiving set of nodes R_p , while the response slots in a cycle are used by nodes in R_p to send back req messages to p . Therefore, there are up to n transmit slots needed. For simplicity, we do not go into detail about how these are set up; we simply assume that nodes in R_p know which response slot to use.

In our p in our pseudocode, $R_p = N(p) \cap C$ and p executes LOCAL BROADCAST(m, p, R_p) in the context of the global broadcast schedule. By this, we mean that a slot in the Probabilistic Sending Phase of LOCAL BROADCAST corresponds to p 's transmit slot in some cycle, the next slot in that same Probabilistic Sending Phase corresponds to p 's transmit slot in the next cycle, and so on. The same thing happens with the Deterministic Sending Phase. In both cases, the response slots are unused. Then in the Probabilistic Nack Phase and Deterministic Nack Phase, the responses are used by each R_p set in the same fashion, where p now listens. By using a response slot, the nodes in R_p send back to p simultaneously as in LOCAL BROADCAST. Note that in the Probabilistic and Deterministic Nack phases, the transmit slots are now unused. Therefore, in each cycle, only the transmit slots or response slots, but not both, are used. For LOCAL BROADCAST running in at most D slots, executing LOCAL BROADCAST in the context of the global broadcast schedule requires at most D cycles. In our pseudocode in Figure 5, we have $D = 25 \cdot (B_0 + 2^{\varphi-1} \ln^{\varphi-1} n)^{\varphi+1}$ in concordance with Lemma 3.5.

We include the detailed proof of completeness for Theorem 1.3 by showing that each correct node

DOS-RESISTANT RELIABLE BROADCAST

- Starting in cycle 1, and ending no later than cycle $D = 25 \cdot (B_0 + 2^{\varphi-1} \ln^{\varphi-1} n)^{\varphi+1}$, node d executes LOCAL BROADCAST(m, d, R_d) and each node $i \in R_d$ commits to the first value it receives from d .

As in [13], $p(x, y)$ commits to m when, through Steps 4 and 5, it receives $t + 1$ COMMIT(q, m) or HEARD(q_2, q_1, m) from node-disjoint paths all lying within a single $(2r + 1) \times (2r + 1)$ area; our analysis shows this occurs in cycle $2yD - 1$. The following step is executed by each node p :

- Starting in cycle $2yD$, and ending no later than cycle $(2y + 1)D - 1$, node $p(x, y)$ performs LOCAL BROADCAST(COMMIT(p, m), p, R_p).

The following steps are executed by each node excluding those nodes in $N(d)$. Any such correct node commits to m when it receives HEARD or COMMIT m from $t + 1$ node-disjoint paths within a single neighborhood.

For $i = 0$ to $r - 1$

- Starting in cycle $2(y - r + i)D$, and ending no later than cycle $2(y - r + i)D + D - 1$, node $p(x, y)$ listens for COMMIT messages by executing LOCAL BROADCAST(COMMIT(q, m), $q(x', y')$, R_q) with each node in row $y' = y - r + i$ in C and where $p \in R_q$.
- Starting in cycle $2(y - r + i)D + D$, and ending no later than cycle $2(y - r + i)D + 2D - 1$, node $p(x, y)$ listens for HEARD messages by executing LOCAL BROADCAST(HEARD(q_2, q_1, m), q_2, R_{q_2}) with each node $q_2 \in B'_p$ in row $y + i$ and where $p \in R_{q_2}$.
- Starting in cycle $2(y - r)D + D$, and ending no later than cycle $2(y - r)D + 2D - 1$, node q_2 sends a HEARD message by executing LOCAL BROADCAST(HEARD(q_2, q_1, m), q_2, R_{q_2}) where q_1, q_2 are sister nodes.

Figure 5: Pseudocode for DOS-RESISTANT RELIABLE BROADCAST.

eventually commits to the correct value m sent by the dealer (the cost analysis is provided later). The following Lemma 3.6 proves the correctness of our protocol in the grid; we emphasize that our argument follows that of [13].

Lemma 3.6. *Assume for each node p , $t < (r/2)(2r + 1)$ nodes in $N(p)$ are Byzantine and used by Carol to disrupt p 's communications for $\beta \leq B_0$ time slots. Let $C = \{\text{Nodes } q \text{ at } (x, y) \mid -r \leq x \leq r \wedge y \geq 0\}$ be a corridor of nodes in this network. Then, DOS-RESISTANT RELIABLE BROADCAST achieves reliable broadcast in C .*

Proof. In [13], it is shown that each node $p(x, y)$ can obtain m by majority filtering on messages from $2t + 1$ node-disjoint paths contained within a single $(2r + 1) \times (2r + 1)$ area since at least $t + 1$ will be m . Our correctness proof is similar; however, we argue along a corridor and show that nodes in the y^{th} row can commit to m by slot $2yD - 1$.

Base Case: Each node in $N(d)$ commits to the correct message m immediately upon hearing it directly from the dealer by cycle D . Therefore, clearly, every node $p(x, y) \in N(d)$ commits by cycle $2yD - 1$.

Induction Hypothesis: Let $-r \leq a \leq r$. If each correct node $p'(x', y') \in N(a, b)$ commits to m by cycle $2y'D - 1$, then each correct node $p(x, y) \in N(a, b + 1) - N(a, b)$ commits to m in cycle $2yD - 1$.

Induction Step: We now show $2t + 1$ connectedness within a single neighborhood and we argue simultaneously about the time required for p to hear messages along these disjoint paths. The node $p(x, y)$ lies in $N(a, b + 1) - N(a, b)$ and can be considered to have location $(a - r + z, b + r + 1)$ where $0 \leq z \leq r$ (the case for $r + 1 \leq z \leq 2r$ follows by symmetry). We demonstrate that there exist $r(2r + 1)$ node-disjoint paths $P_1, \dots, P_{r(2r+1)}$ all lying within the same neighborhood and that the synchronization prescribed by our protocol is correct:

One-Hop Paths: the set of nodes $A_p = \{q(u, v) \mid (a - r) \leq u \leq (a + z) \text{ and } (b + 1) \leq v \leq (b + r)\}$ lie in $N(a, b)$ and neighbor p . Therefore, there are $r(r + z + 1)$ paths of the form $q \rightarrow p$ where $q \in A_p$.

By their position relative to $p(x, y)$, each correct node $q(u, v) \in A_p$ is such that $v = y - r + c$ for some fixed $c \in \{0, \dots, r - 1\}$. Therefore, by the induction hypothesis, q commits to m by cycle $2(y - r + c)D - 1$. By the protocol, $q(u, v)$ sends COMMIT messages using LOCAL BROADCAST in cycle $2vD = 2(y - r + c)D$ until cycle $2(v + 1)D - 1 = (2(y - r + c) + 1)D - 1$ at the latest. By the protocol, $p(x, y)$ listens for COMMIT messages from q starting in cycle $2(y - r + c)D$ until $(2(y - r + c) + 1)D - 1$ at the latest; note that p listens to many executions of LOCAL BROADCAST containing HEARD messages, but we focus on this particular one from q . Therefore, p and q are synchronized in the execution of LOCAL BROADCAST and p will receive q 's message by cycle $(2(y - r + c) + 1)D - 1 = (2(b + c + 1) + 1)D - 1$ at the latest. Since this occurs for all nodes in A_p , node p has received all COMMIT messages from A_p by cycle $(2(y - 1) + 1)D - 1 = (2(b + r) + 1)D - 1 \leq 2(b + r + 1)D - 1 = 2yD - 1$.

Two-Hop Paths: consider the sets $B_p = \{q(u, v) \mid (a + z + 1) \leq u \leq (a + r) \text{ and } (b + 1) \leq v \leq (b + r)\}$ and $B'_p = \{q'(u', v') \mid (a + z + 1 - r) \leq u' \leq (a) \text{ and } (b + r + 1) \leq v' \leq (b + 2r)\}$. The set B'_p is obtained by shifting left by r units and up by r units. Recall that there is a one-to-one mapping between the nodes in B_p and the nodes in B'_p ; these are sister nodes. There are $r(r - z)$ paths of the form $q \rightarrow q' \rightarrow p$ where q and q' are sister nodes. These sets, and the notion of sister nodes, are illustrated in Figure 4.

Consider a correct node $q(u, v) \in B_p$ and its sister node $q'(u', v') \in B'_p$ where $v' = v + r$ by definition. Again, given the location of $q(u, v)$ relative to $p(x, y)$, we have $v = y - r + c$ for some fixed $c \in \{0, \dots, r - 1\}$. By the induction hypothesis, q commits to m by cycle $2vD - 1$. Then by DOS-RESISTANT RELIABLE BROADCAST, q sends a COMMIT message using LOCAL BROADCAST in cycle $2vD = 2(y - r + c)D$ until cycle $2(v + 1)D - 1 = (2(y - r + c) + 1)D - 1$ at the latest. Again, this is the particular execution of LOCAL BROADCAST between q and q' ; q performs others. By DOS-RESISTANT RELIABLE BROADCAST, $q'(u', v')$ receives COMMIT messages from q using LOCAL BROADCAST starting in cycle $2(v' - r + c)D = 2vD = 2(y - r + c)D$ and ending no later than cycle $2(v' - r + c + 1)D - 1 = 2(v + 1)D - 1 = (2(y - r + c) + 1)D - 1$.

By the above, each node $q'(u', v') \in B'_p$ is able to start sending a HEARD message using LOCAL BROADCAST in cycle $2(v' - r)D + D$ and ending no later than cycle $2(v' - r)D + 2D - 1$. Starting in cycle $2(y - r + c)D + D$, node $p(x, y)$ uses LOCAL BROADCAST to listen for a HEARD message from $q'(u', v')$ where $v' = y + c$. Therefore, p is listening to q' starting in $2(y - r + c)D + D = 2(v' - r)D + D$ and ending no later than $2(v' - r)D + 2D - 1$; p and q' are synchronized. Therefore, p receives all HEARD messages by cycle $2(v' - r)D + 2D - 1$ when $v' = y + r - 1$ (the node at the top row of B'_p); that is, by cycle $2(y - 1)D + 2D - 1 = 2yD - 1$.

Therefore, a total of $r(r + z + 1) + r(r - z) = r(2r + 1)$ node-disjoint paths from $N(a, b)$ to $PN(a, b)$ exist, all lying in in a single neighborhood $N(a, b + r + 1)$. For an adversary corrupting $t < (r/2)(2r + 1)$ nodes, a correct node can majority filter to obtain m . Furthermore, we have shown that any $p(x, y) \in N(a, b + 1)$

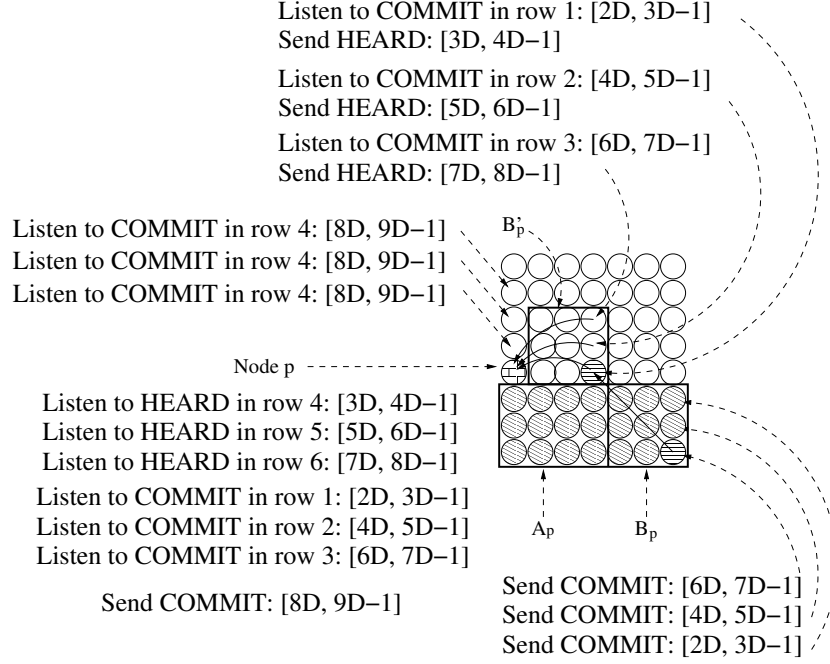


Figure 6: An example of some steps of the protocol for $r = 3$. The node in row 4 highlighted with the horizontal lines in the B'_p listens to a COMMIT message from its sister node in B_p by partaking in LOCAL BROADCAST as a receiver from cycle $2D$ to cycle $3D - 1$. Then, in cycle $3D$ to cycle $4D - 1$, that node uses LOCAL BROADCAST to send a HEARD message to p who is listening in this execution of LOCAL BROADCAST from cycle $3D$ to cycle $4D - 1$. The listening for p for each row is described on the left; note the synchronization. We also illustrate that those nodes above p will be listening for p 's COMMIT message using LOCAL BROADCAST at the appropriate time.

executes LOCAL BROADCAST $r(2r + 1) = O(r^2) = O(t)$ times in order to receives all COMMIT and HEARD messages by cycle $2yD - 1$. Therefore, p can commit to the correct message by cycle $2yD - 1$; this concludes the induction. \square

Finally, we reiterate that proving reliable broadcast in a corridor is sufficient as the entire grid (with the exception of the boundary of width less than r in the case of a finite grid) can be covered piecewise by such corridors.

3.6 Reliable Broadcast in General Topologies

We examine the grid model previously because it features in previous literature on jamming-resistant reliable broadcast [1, 11, 15]. In this section, we present our results for reliable broadcast on an arbitrary graph $G = (V, E)$. Pelc and Peleg [52] examine a generalization of the t -locally bounded fault model; that is, where each node contains at most t Byzantine nodes within its neighborhood. Specifically, they examine the broadcast protocol of Koo [37], which the authors call the *Certified Propagation Algorithm* (CPA), with the aim of establishing conditions for which it achieves reliable broadcast under arbitrary graphs in contrast to the grid model. CPA does not always achieve optimal fault tolerance; for example, it cannot tolerate the optimal number of faults $t = (r/2)(2r + 1) - 1$ in the grid as we do above. However, we address CPA because its generality is powerful.

Again, CPA requires that all nodes obey a global broadcast schedule (i.e. there is no jamming adversary). Pelc and Peleg [52] define $X(p, d)$ to be the number of nodes in p 's neighborhood $N(p)$ that are closer to d than p and then introduce the parameter $X(G) = \min\{X(p, d) \mid p, d \in V, (p, s) \notin E\}$. To reiterate, one of their main results is that, for *any* graph G with dealer d such that $t < X(G)/2$, CPA achieves reliable broadcast. For our purposes, define for each node p the set of nodes $X(p)$ to be those $X(p, d)$ nodes closer to the dealer than p . Clearly, it is possible to identify $X(p)$ in polynomial time and so we observe:

Observation 3.7. *If the topology of G and the location of d is known to all nodes, then each node p can calculate $X(p)$.*

3.6.1 A Favourable Protocol in General Topologies

The pseudocode for CPA is given in Figure 7. Note that, unless a node is sending in the slot allotted to it by the global broadcast schedule, or it has terminated, it is perpetually listening. We aim to remove this wasteful listening by synchronizing the sending and listening of nodes.

In the context of CPA, we call a single iteration of the global broadcast schedule a *broadcast round* (we use cycles again later on when we modify CPA). Throughout, assume that time is measured from when the dealer first broadcasts m in broadcast round 0. Under CPA, regardless of the worst-case delay imposed by the adversary, there is a broadcast round where p must have received at least $t + 1$ messages from distinct correct nodes in $X(p)$ allowing p to commit to m ; denote this broadcast round by s_p . Note, that in any execution of reliable broadcast, p may actually be able to commit before broadcast round s_p , but s_p is the maximum broadcast round in which p is guaranteed to have all the information it needs to commit to m regardless of how the adversarial nodes behave.

Since a correct node $u \in N(d)$ accepts what it hears from the dealer d immediately, and d 's broadcast round is 0, $s_u = 1$. For nodes not in $N(d)$, the situation is slightly more complicated. In the grid, for node $p(x, y)$, we were able to compute s_p explicitly (in terms of cycles) as $2yD - 1$ in the corridor C (see proof of Lemma 3.6). Here, unlike with the grid, we cannot specify s_p explicitly for any graph G because it is dependent on the topology; however, by the correctness of CPA, every node eventually commits and so s_p must exist for each node p . In fact, our protocol based on CPA is simpler than that in the grid because the protocol of Bhandari and Vaidya [13] uses HEARD messages (which makes the synchronization tedious), while CPA uses only COMMIT messages.

For a *fixed* G whose topology is known to all nodes (including knowledge of where the dealer d is situated), each node p can calculate s_p . This is done by simulating the propagation of m using CPA. In this simulation, each node p has the maximum $t = X(G)/2 - 1$ Byzantine nodes in $X(p)$ and these Byzantine nodes send their faulty messages prior to the $t + 1$ correct responses in order delay propagation of m for as long as possible. By assuming that every $X(p)$ has the maximum number of Byzantine nodes, the actual placement of the Byzantine nodes in G does not affect the worst-case broadcast time s_p . In tracing this propagation, any node can calculate s_p for any node p .

Now, consider the following minor modifications to CPA: (1) each correct node p only listens to $q \in X(p)$ in broadcast round $s_q + 1$, and (2) each correct node p only sends its commit message in broadcast round $s_p + 1$. In all other slots, a node p is sleeping. This is a minor modification of CPA; call it CPA_0 . These modifications synchronize the sending/listening and allow nodes to otherwise sleep instead of perpetually listening as in CPA. The pseudocode for CPA_0 is given in Figure 8. While it is fairly clear that this modification does not affect correctness, we state it formally for completeness.

Lemma 3.8. *If CPA achieves reliable broadcast, then CPA_0 achieves reliable broadcast.*

Proof. For every node p , assume $X(p)$ has the maximum $t = X(G)/2 - 1$ Byzantine nodes and that these Byzantine nodes all send their messages to p ahead of the correct nodes in $X(p)$ according to the broadcast

Certified Propagation Algorithm ([37] and [52])

- The dealer d sends the message to all of its neighbors and terminates.
- For a correct node $u \in N(d)$, upon receiving m from d it commits to m , node u announces this commitment of its neighbors and terminates.
- If a node is not a neighbor of the source, then upon receiving $t + 1$ copies of m from $t + 1$ distinct neighbors, it commits to m , and announces this commitment to its neighbors and terminates.

Figure 7: Pseudocode for the Certified Propagation Algorithm (CPA)

CPA₀

- In broadcast round 0, the dealer d sends the message to all of its neighbors and terminates.
- For a correct node $u \in N(d)$, u listens in broadcast round 0 and accepts m as correct, announces this commitment to its neighbors in broadcast round 1, and terminates.
- If a node p is not a neighbor of the source, then p listens to each neighbor $q \in X(p)$ in broadcast round $s_q + 1$; otherwise, p sleeps. Upon receiving $t + 1$ copies of m from $t + 1$ distinct neighbors in $X(p)$, it accepts m as correct, announces this commitment to its neighbors in broadcast round $s_p + 1$, and terminates.

Figure 8: Pseudocode for CPA₀.

schedule. Pelc and Peleg [52] showed that CPA is correct in this situation (their result is independent of any particular ordering of sending in the broadcast schedule; that is, CPA remains correct if Byzantine nodes always send first). Therefore, in this case, each correct node p would receive a commitment from $q \in X(p)$ in broadcast round $s_q + 1$ and node p would announce its commitment in round $s_p + 1$. This is exactly what happens in CPA₀ with nodes sleeping otherwise. Therefore, if CPA achieves reliable broadcast, then so does CPA₀. \square

Define a cycle as done before in the grid. In Figure 9, we provide pseudocode for a fair and favourable reliable broadcast algorithm FCPA that tolerates the jamming adversary described in Theorem 1.3.

Lemma 3.9. *Assume CPA achieves reliable broadcast on a graph G . Then FCPA guarantees reliable broadcast on G .*

Proof. Using FCPA, we claim that every correct node p can commit by cycle $s_p \cdot D$. To prove this, assume the opposite: that some node p does not commit to the correct value by cycle $s_p \cdot D$. Then, there is some correct node $q \in X(p)$ that: (1) could not commit to a message by time slot $s_q \cdot D$ (and could not send p a commitment message), or (2) committed to a wrong message (and sent that wrong message to p). Note that the time for any node p' to send its commit message to a node p'' is at most D cycles by Lemma 3.5. Therefore, if q cannot commit (or commits to the wrong value) by $s_q \cdot D$ in FCPA, then q cannot commit

FCPA

- The dealer d sends the message to all of its neighbors using LOCAL BROADCAST($m, d, N(d)$) and terminates after at most $D = 25 \cdot (B_0 + 2^{\varphi-1} \ln^{\varphi-1} n)^{\varphi+1}$ cycles.
- If node $u \in N(d)$, then upon receiving m from d via LOCAL BROADCAST($m, d, N(d)$), it accepts m as correct, announces this commitment to its neighbors in cycle D , and terminates.
- If a node p is not a neighbor of the source, then p listens to each neighbor $q \in X(p)$ via LOCAL BROADCAST($m, q, N(q)$) starting in cycle $s_q \cdot D + 1$ and ending by $(s_q + 1) \cdot D$; otherwise, p sleeps. Upon receiving $t + 1$ copies of m in this fashion from $t + 1$ distinct neighbors in $X(p)$, it accepts m as correct, announces this commitment to its neighbors using LOCAL BROADCAST($m, p, N(p)$) in broadcast cycle $s_p \cdot D + 1$, and terminates by cycle $(s_p + 1) \cdot D$.

Figure 9: Pseudocode for FCPA.

by cycle s_q in CPA₀; therefore, CPA₀ fails to achieve reliable broadcast. However, if CPA₀ fails to achieve reliable broadcast, then by the contrapositive of Lemma 3.8, this contradicts the assumption that CPA achieves reliable broadcast. \square

3.6.2 Favourability Analysis

Finally, the following analysis proves favourability, both for the grid and for general topologies:

Theorem 1.3 —Cost Analysis: In both of our protocols, each correct node p partakes in an execution of LOCAL BROADCAST $O(t)$ times as a sender and receiver; let k denote the total number of such executions. For the i^{th} such execution, let τ_i be the number of slots for which the adversary is active for $i = 1, \dots, k$. Denote the adversary's total active time by $\beta = \sum_{i=1}^k \tau_i \leq B_0$. Consider two cases:

Case I: Assume the adversary is active for a total of $\beta = \sum_{i=1}^k \tau_i = O(t \ln^{\varphi+1} t)$ slots over all k executions of LOCAL BROADCAST involving p . For each execution, p incurs $O(\tau_i^{\varphi-1} \ln t + \ln^{\varphi} t)$ cost in expectation by Theorem 1.2. Therefore, over $k = O(t)$ executions, p 's expected total cost is $O((\sum_{i=1}^k \tau_i^{\varphi-1}) \ln t + t \ln^{\varphi} t) = O((\sum_{i=1}^k \tau_i) \ln t + t \ln^{\varphi} t) = O(\beta \ln t + t \ln^{\varphi} t) = O(t \ln^{\varphi+2} t)$.

Case II: Otherwise, $\beta = \sum_{i=1}^k \tau_i = \omega(t \ln^{\varphi+1} t)$. By Jensen's inequality for concave functions, for a concave function f , $f(\frac{1}{k} \sum_{i=1}^k \tau_i) \geq \frac{1}{k} \sum_{i=1}^k f(\tau_i)$. Since $f(\tau) = \tau^{\varphi-1}$ is concave, it follows that $\sum_{i=1}^k \tau_i^{\varphi-1} \leq k(\frac{1}{k} \sum_{i=1}^k \tau_i)^{\varphi-1} = k^{2-\varphi} (\sum_{i=1}^k \tau_i)^{\varphi-1}$. Therefore, the total expected cost to p over $k = O(t)$ executions is $O((\sum_{i=1}^k \tau_i^{\varphi-1}) \ln t) + O(t \ln^{\varphi} t) = O(t^{2-\varphi} (\sum_{i=1}^k \tau_i)^{\varphi-1} \ln t) + O(t \ln^{\varphi} t) = O(t^{(2-\varphi)\beta^{\varphi-1} \ln t + t \ln^{\varphi} t) = o(\beta)$. Therefore, p 's expected cost is less than that of the adversary.

Substituting $t = O(r^2)$ into the above analysis yields the favourability result in the grid and, together, gives our result for the grid model in Theorem 1.3. \square

Therefore, while the adversarial nodes incur a cost of β , reliable broadcast is possible with each correct node incurring a cost of $o(\beta)$ for β sufficiently large; this yields Theorem 1.3.

3.6.3 The Las Vegas Guarantee in Multi-Hop WSNs

In addition to the rationale given in Section 1.3, the Las Vegas property is also valuable in multi-hop sensor networks for the following reason. Let n be the number of devices within transmitting distance of a device, and let N be the total number of devices in the network. Monte Carlo protocols that succeed with high probability in n are possible. However, typically, $n \ll N$ and messages will traverse a chain of multiple hops; consider $\Omega(N)$ hops. *Consider a failure probability for a single hop that is small, but non-zero, such as $\Theta(n^{-c})$ for some constant $c > 0$, or even $\Theta(2^{-n})$, then communication fails along the chain with constant probability.* Alternatively, we might achieve protocols that succeed with high probability in N . However, in large networks, N may not be known *a priori*. Furthermore, achieving a high probability guarantee in N typically involves $\Omega(\log N)$ operations which, for large N , may be too costly. Therefore, by devising Las Vegas protocols, we avoid assumptions that are problematic given that $n \ll N$.

We note that transmission over the wireless medium is subject to error due to radio-irregularity and gray-zone effects. Does this reduce the utility of our Las Vegas guarantee? In many cases, we argue that it does not. Under fair weather conditions, the percentage of successfully received packets is nearly 100% up to a distance threshold exceeding 25 meters in the case of the MICA2DOT mote [2]. Other experimental studies have shown that communication is reliable so long as the signal-to-interference-plus-noise-ratio exceeds a threshold value [63, 64]; therefore, using a transmission power above this threshold yields reliable communication. Other experimental studies on the packet reception rate, which closely approximates the probability of successfully receiving a packet between two neighbouring nodes, is perfect up to a fixed distance [77]. Therefore, for an appropriate transmission power in dense sensor networks, communication over the wireless medium should not undermine our Las Vegas guarantee.

4 Application 2: Application-Level DDoS Attacks

Typically in application-level DDoS attacks, a number of compromised clients, known collectively as a *botnet*, are employed to overwhelm a server with requests. These botnets have become commercialized with operators (“botmasters”) renting out time to individuals for the purposes of launching attacks [23, 39].

We assume a model of botnet attacks similar to that described by Walfish *et al.* [69]. In this model, a request is cheap for a client to issue, expensive for the server to service, and all requests incur the same computational cost (heterogeneous requests can likely be handled as in [69]). There is a high-capacity communication channel and the crucial bottleneck is the server’s inability to process a heavy request load.

The client rate is g requests per second. The aggregate botnet rate is R requests per second and this is assumed to be both relatively constant and the botnet’s maximum possible rate. If the server is overloaded, it randomly drops excess requests. In this case, the good clients only receive a fraction $g/(g + R)$ of the server’s resources; it is assumed that $R \gg g$ so that $g/(g + R)$ is very small.

Walfish *et al.* [69] propose a protocol SPEAK-UP for resisting DDoS attacks by having clients increase their sending rate such that their aggregate bandwidth G is on the same order as that of R . Since botnet machines are assumed to have already “maxed-out” their available bandwidth in attacking, SPEAK-UP greatly increases the chance that the server processes a legitimate request since $G/(G + R) \gg g/(g + R)$. A crucial component of SPEAK-UP is a front-end to the server called the “thinner” which controls which requests are seen by the server and asks a client to retry her request if it was previously dropped. Here, we apply our results for Case 2 to the interactions between the client and the thinner.

Lastly, we note that there are a number of questions that naturally arise when considering the scheme of Walfish *et al.* and our adaptation here. How will the increased traffic from the clients affect the overall performance of the network? For what size of botnet is this approach applicable? These questions, and many others, are addressed in the original paper by Walfish *et al.* [69] and we refer the interested reader to this work.

DOS-RESISTANT CLIENT-SERVER COMMUNICATION for round $i \geq 2$

Send Phase: For each of the 2^{2i} slots do

- The client sends her request with probability $2/2^i$.
- The server (via the thinner) admits listens with probability $2/2^i$.

Nack Phase:

- The server sends back the requested data.
- The client listens.

If the client receives her data, she terminates; otherwise, the thinner tells her to retry in the next round.

Figure 10: Pseudocode for the application of Case 2 of our 3-Player Scenario to the client-server scenario.

4.1 Our Protocol

We employ Case 2 of our 3-PLAYER SCENARIO PROTOCOL to achieve a SPEAK-UP-like algorithm with provable guarantees. Bandwidth (upstream and downstream rates in bits per second) is our measure of cost and, as such, our results should be interpreted as quantifying the expected upstream bandwidth required by the client and the expected downstream bandwidth with which the server should be provisioned. Using bandwidth as a form of currency has been previously employed by the research community [29, 61, 69]. Our pseudocode is given in Figure 10.

Note that the Nack Phase is simplified due to the fact that attacks do not occur in this phase for Case 2 of the 3-PLAYER SCENARIO PROTOCOL. Like [69], our protocol is suitable for applications where there is no pre-defined clientele (so the server cannot traffic filter) and the clientele can be non-human (so “proof-of-humanity” tests cannot be relied upon solely). Unlike the wireless domain, we do not address reactive adversaries. Determining when a player is sending over the wire in order to control when its traffic arrives at the targeted player seems beyond the capability of a realistic attacker.

The client plays the role of Alice where the message is a request; the server plays the role of Bob. This application falls into Case 2 of Theorem 1.1: a DDoS attack targets the server while communications from the server to the clients are not disrupted. The client and server are assumed to be synchronized such that they always agree on the current round and a maximum round number is set *a priori*. Such synchronization is certainly possible over Internet-connected machines and the maximum round value should be set to account for the level of DDoS resistance the participants wish to have; for most attacks, R is in the low hundreds of Mbits/second [60]. We now provide an overview of our protocol.

Send Phase: Each Send Phase occurs over a uniform and fixed duration Δ ; for simplicity, we set $\Delta = 1$ second, and the slot length changes in each round appropriately. The client sends in each slot with probability $2/2^i$ with an expected 2^{i+1} upstream bits per second. The server listens in each slot with probability $2/2^i$ for an expected 2^{i+1} downstream bits per second. If the received traffic substantially exceeds 2^{i+1} , requests are dropped; probabilistic listening and traffic measurement on the server side can be performed by the thinner [69].

Note that in each round, the client increases her sending rate in the Send Phase to “speak up”. Any correct client that reaches its bandwidth limit remains at this limit for the duration of the protocol. When the maximum round number is reached, the clients maintain their sending rate until the thinner informs them that the attack has ended. For the purposes of analysis, a blocked slot occurs when Carol overwhelms

the server with requests and the client's request is dropped in that slot. Define a send-blocked phase as one where Carol blocks at least $2^{2i}/2$ slots; therefore, Carol uses an upstream bandwidth of *at least* $2^{2i}/2$ bits per second. As in [69], if the thinner drops a request, it immediately asks the client to retry in the next round.

Nack Phase: The server does not increase its sending rate per round (only the client speaks up) since there are no attacks in the Nack Phase for Case 2. This simplifies the Nack Phase as mentioned in Section 2 in our discussion of Nack Failures; the server simply returns the requested data to the client at some reasonable rate.

We assume upstream and downstream bandwidth are capped; this is true of residential Internet packages, as well as hosted services. In the case of residential service, upstream bandwidth is scarcer than downstream bandwidth, while servers are generally well-provisioned for both; this can be reflected in our cost constants. By Case 2 of Theorem 1.1 we have:

Corollary 4.1. *If Carol uses bandwidth R to attack, then the client's request is eventually serviced, and the expected bandwidth (upstream and downstream) used by the client and the server is $O(R^{0.5})$.*

Bob can represent multiple good clients. We assume the same synchronization with the server; however, clients joining at different times are informed by the thinner of the current round. In order to be guaranteed *some* of the server's resources, the clients' expected aggregate bandwidth is $G = \Omega(R^{0.5})$. Therefore, our result quantifies the minimum expected aggregate upstream bandwidth for clients and the expected downstream bandwidth for the server required to ensure that total censorship is averted; in contrast, SPEAK-UP cannot make such a guarantee.

Of course, making it costly for the adversary to achieve an attack that results in zero throughput does not solve the problem of DDoS attacks by any means. Ideally, we would like to see a more result that quantifies cost versus performance more generally and this is an area for future work. However, we do point out that our result might be useful for applications where a critical update or warning must be disseminated, and delivery to even a handful of clients is sufficient since they may then share it with others (via multicast, peer-to-peer distribution, etc.).

As with SPEAK-UP, the probability of legitimate request being serviced is still $G/(G + R)$. In addition to admitting an analysis, our iterative approach of geometrically increasing the aggregate bandwidth should mitigate attempts by Carol at launching short duration DDoS attacks in order to provoke a steep and disruptive traffic increase from correct clients. In the context of Section 1.4, our protocol is fair in that the *aggregate* requirements of the bandwidth constrained clients is asymptotically equal to that of the well-provisioned server. Restating our result above in the context of multiple clients yields Theorem 1.4.

Finally, in order to achieve the same level of denial of service against a server that is defended by our protocol, Carol must procure a much larger botnet in order to obtain the necessary bandwidth; however, this comes at a cost. For example, one study found the cost of a single bot to be between \$2 and \$25 [23]. Therefore, since Carol's bandwidth requirements increase quadratically, her monetary costs increase significantly with the use of our protocol.

5 Conclusion

We have examined an abstract model of conflict over a communication channel. In the 3-Player Scenario, we remark that there is an $O(1)$ up-front cost per execution of our protocol when there are no send- or nack-blocking attacks. Similarly, there are small up-front costs for our other favorable protocols. This is the (tolerable) price for communication in the presence of a powerful adversary, even if that adversary is not necessarily very active. The golden ratio arises naturally from our analysis, and its appearance in this adversarial setting is interesting; an important open question is whether $\Omega(B^{\varphi-1} + 1)$ cost is necessary.

Also of interest is determining whether there are resource-competitive algorithms for other types of problems. An interesting starting point might be the problem of conflict over dissemination of an idea in a social network, using the models of Kempe *et al.* [33]. Another challenge is whether or not more general and powerful results are possible regarding DDoS attacks in the client-server scenario. Finally, while we have applied resource-competitive analysis to fundamental communication scenarios, we are interested using this approach to address other problems from the area of distributed computing such as Byzantine agreement and consensus in the wired domain.

Acknowledgements: We thank Martin Karsten, Srinivasan Keshav, and James Horey for their valuable comments.

References

- [1] Dan Alistarh, Seth Gilbert, Rachid Guerraoui, Zarko Milosevic, and Calvin Newport. Securing Your Every Bit: Reliable Broadcast in Byzantine Wireless Networks. In *Proceedings of the 22nd Symposium on Parallelism in Algorithms and Architectures (SPAA)*, pages 50–59, 2010.
- [2] Giuseppe Anastasi, A. Falchi, Andrea Passarella, Marco Conti, and Enrico Gregori. Performance Measurements of Motes Sensor Networks. In *Proceedings of the 7th ACM International Symposium on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWiM)*, pages 174–181, 2004.
- [3] Nils Aschenbruck, Elmar Gerhards-Padilla, and Peter Martini. Simulative Evaluation of Adaptive Jamming Detection in Wireless Multi-hop Networks. In *Proceedings of the 30th International Conference on Distributed Computing Systems Workshops*, pages 213–220, 2010.
- [4] Farhana Ashraf, Yih-Chun Hu, and Robin Kravets. Demo: Bankrupting the Jammer. In *Proceedings of the 9th International Conference on Mobile Systems, Applications, and Services (MobiSys)*, 2011.
- [5] Tuomas Aura, Pekka Nikander, and Jussipekka Leiwo. DOS-resistant Authentication with Client Puzzles. In *8th International Workshop on Security Protocols*, pages 170–177, 2000.
- [6] Baruch Awerbuch, Andrea Richa, and Christian Scheideler. A Jamming-Resistant MAC Protocol for Single-Hop Wireless Networks. In *Proceedings of the 27th Symposium on the Principles of Distributed Computing (PODC)*, pages 45–54, 2008.
- [7] Joe Bardwell. Converting Signal Strength Percentage to dBm Values, 2002.
- [8] Emrah Bayraktaroglu, Christopher King, Xin Liu, Guevara Noubir, Rajmohan Rajaraman, and Bishal Thap. On the Performance of IEEE 802.11 under Jamming, 2008.
- [9] BBC. Anonymous Hacktivists Say Wikileaks War to Continue. www.bbc.co.uk/news/technology-11935539, 2010.
- [10] Marin Bertier, Anne-Marie Kermarrec, and Guang Tan. Brief Announcement: Reliable Broadcast Tolerating Byzantine Faults in a Message-Bounded Radio Network. In *Proceedings of the 22nd International Symposium on Distributed Computing (DISC)*, pages 516–517, 2008.
- [11] Marin Bertier, Anne-Marie Kermarrec, and Guang Tan. Message-Efficient Byzantine Fault-Tolerant Broadcast in a Multi-Hop Wireless Sensor Network. In *Proceedings of the 30th International Conference on Distributed Computing Systems (ICDCS)*, pages 408–417, 2010.

- [12] Vartika Bhandari and Nitin H. Vaidya. On Reliable Broadcast in a Radio Network. In *Proceedings of the 24th Annual ACM Symposium on Principles of Distributed Computing (PODC)*, pages 138–147, 2005.
- [13] Vartika Bhandari and Nitin H. Vaidya. On Reliable Broadcast in a Radio Network: A Simplified Characterization. Technical report, CSL, UIUC, May 2005.
- [14] Vartika Bhandari and Nitin H. Vaidya. Reliable Broadcast in Wireless Networks with Probabilistic Failures. In *Proceedings of the International Conference on Computer Communications (INFOCOM)*, pages 715–723, 2007.
- [15] Vartika Bhandhary, Jonathan Katz, Chiu-Yuen Koo, and Nitin Vaidya. Reliable Broadcast in Radio Networks: The Bounded Collision Case. In *Proceedings of the 25th Annual ACM Symposium on Principles of Distributed Computing (PODC)*, pages 258 – 264, 2006.
- [16] John Cox. For Wireless Sensor Nets, Reality Sets In. <http://features.techworld.com/networking/1863/for-wireless-sensor-nets-reality-sets-in/>, 2005.
- [17] Crossbow. Mica2 Wireless Measurement System. [https:// www.eol.ucar.edu/ rtf/facilities/ isa/internal/ CrossBow/ DataSheets/ mica2.pdf](https://www.eol.ucar.edu/rtf/facilities/isa/internal/CrossBow/DataSheets/mica2.pdf).
- [18] Crossbow. MICAz Wireless Measurement System. [http:// courses.ece. ubc.ca/ 494/files/ MI- CAz_Datasheet. pdf](http://courses.ece.ubc.ca/494/files/MICAz_Datasheet.pdf).
- [19] Jing Deng, Pramod K. Varshney, and Zygmunt J. Haas. A New Backoff Algorithm for the IEEE 802.11 Distributed Coordination Function. In *Communication Networks and Distributed Systems Modeling and Simulation*, pages 215–225, 2004.
- [20] Shlomi Dolev, Seth Gilbert, Rachid Guerraoui, Fabian Kuhn, and Calvin Newport. The Wireless Synchronization Problem. In *Proceedings of the 28th ACM Symposium on Principles of Distributed Computing*, PODC ’09, pages 190–199, 2009.
- [21] Shlomi Dolev, Seth Gilbert, Rachid Guerraoui, and Calvin Newport. Gossiping in a Multi-channel Radio Network: An Oblivious Approach to Coping with Malicious Interference. In *Proceedings of the 21st International Symposium on Distributed Computing (DISC)*, pages 208–222, 2007.
- [22] Shlomi Dolev, Seth Gilbert, Rachid Guerraoui, and Calvin Newport. Secure Communication over Radio Channels. In *Proceedings of the 27th ACM Symposium on Principles of Distributed Computing (PODC)*, pages 105–114, 2008.
- [23] Jason Franklin, Vern Paxson, Adrian Perrig, and Stefan Savage. An Inquiry into the Nature and Causes of the Wealth of Internet Miscreants. In *14th ACM Conference on Computer and Communications Security*, pages 375–388, 2007.
- [24] Saurabh Ganeriwal, Christina Pöpper, Srdjan Čapkun, and Mani B. Srivastava. Secure Time Synchronization in Sensor Networks. *ACM Transactions on Information and System Security*, 11(23), 2008.
- [25] Lee Garber. Denial-of-Service Attacks Rip the Internet. *Computer*, 33(4):12–17, 2000.
- [26] Seth Gilbert, Rachid Guerraoui, Dariusz Kowalski, and Calvin Newport. Interference-Resilient Information Exchange. In *Proceedings of the International Conference on Computer Communications (INFOCOM)*, pages 2249–2257, 2009.

- [27] Seth Gilbert, Rachid Guerraoui, and Calvin C. Newport. Of malicious motes and suspicious sensors: On the efficiency of malicious interference in wireless networks. In *Proceedings of the International Conference On Principles Of Distributed Systems (OPODIS)*, pages 215–229, 2006.
- [28] Virgil D. Gligor. Guaranteeing Access in Spite of Distributed Service-Flooding Attacks. In *Security Protocols Workshop*, 2003.
- [29] Carl A. Gunter, Sanjeev Khanna, Kaijun Tan, and Santosh Venkatesh. DoS Protection for Reliably Authenticated Broadcast. In *Proceedings of the 11th Networks and Distributed System Security Symposium (NDSS)*, 2004.
- [30] Wendi Rabiner Heinzelman, Anantha Chandrakasan, and Hari Balakrishnan. Energy-Efficient Communication Protocol for Wireless Microsensor Networks. In *Proceedings of the 33rd Hawaii International Conference on System Sciences (HICSS)*, pages 3005–3014, 2000.
- [31] Ari Juels and John Brainard. Client Puzzles: A Cryptographic Countermeasure Against Connection Depletion Attacks. In *Networks and Distributed Security Systems (NDSS)*, pages 151–165, 1999.
- [32] Chris Karlof, Naveen Sastry, and David Wagner. TinySec: A Link Layer Security Architecture for Wireless Sensor Networks. In *Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems (SenSys)*, pages 162–175, 2004.
- [33] D. Kempe, J. Kleinberg, and É. Tardos. Maximizing the spread of influence through a social network. In *Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 137–146, 2003.
- [34] Valerie King, Cynthia Phillips, Jared Saia, and Maxwell Young. Sleeping on the Job: Energy-Efficient and Robust Broadcast for Radio Networks. In *Proceedings of the 27th ACM symposium on Principles of Distributed Computing (PODC)*, pages 243–252, 2008.
- [35] Valerie King, Cynthia Phillips, Jared Saia, and Maxwell Young. Sleeping on the Job: Energy-Efficient and Robust Broadcast for Radio Networks. *Accepted to Algorithmica*, 2010.
- [36] Valerie King, Jared Saia, and Maxwell Young. Conflict on a Communication Channel. In *Proceedings of the 30th Symposium on Principles of Distributed Computing (PODC)*, pages 277–286, 2011.
- [37] Chiu-Yuen Koo. Broadcast in Radio Networks Tolerating Byzantine Adversarial Behavior. In *Proceedings of the 23rd Annual ACM Symposium on Principles of Distributed Computing (PODC)*, pages 275–282, 2004.
- [38] Yee Wei Law, Jeroen Doumen, and Pieter Hartel. Survey and Benchmark of Block Ciphers for Wireless Sensor Networks. *ACM Transactions on Sensor Networks*, 2(1):65–93, 2006.
- [39] Michael Lesk. The New Front Line: Estonia under Cyberassault. *IEEE Security and Privacy*, 5(6):76–79, 2007.
- [40] Yuan Li, Wei Ye, and John Heidemann. Energy and Latency Control in Low Duty Cycle MAC Protocols. In *Proceedings of the IEEE Wireless Communications and Networking Conference*, pages 676–682, 2005.
- [41] Guolong Lin and Guevara Noubir. On Link Layer Denial of Service in Data Wireless LANs. *Wireless Communications & Mobile Computing*, 5(3):273–284, 2005.

- [42] Donggang Liu and Peng Ning. Multi-Level μ TESLA: Broadcast Authentication for Distributed Sensor Networks. *ACM Transactions in Embedded Computing Systems*, 3:800–836, 2004.
- [43] Xin Liu, Guevara Noubir, Ravi Sundaram, and San Tan. SPREAD: Foiling Smart Jammers using Multi-layer Agility. In *Proceedings of the International Conference on Computer Communications (INFOCOM)*, pages 2536–2540, 2007.
- [44] Ewen MacAskill. WikiLeaks Website Pulled by Amazon after U.S. Political Pressure. www.guardian.co.uk/media/2010/dec/01/wikileaks-website-cables-servers-amazon, 2010.
- [45] M. H. Manshaei, Q. Zhu, T. Alpcan, T. Başar, and J.-P. Hubaux. Game Theory Meets Network Security and Privacy. Technical Report EPFL-REPORT-151965, Ecole Polytechnique Fédérale de Lausanne (EPFL), 2010.
- [46] Dominic Meier, Yvonne Anne Pignolet, Stefan Schmid, and Roger Wattenhofer. Speed Dating Despite Jammers. In *Proceedings of the 5th IEEE International Conference on Distributed Computing in Sensor Systems (DCOSS)*, pages 1–14, 2009.
- [47] William G. Morein, Angelos Stavrou, Debra L. Cook, Angelos D. Keromytis, Vishal Misra, and Dan Rubenstein. Using Graphic Turing Tests to Counter Automated DDoS Attacks against Web Servers. In *Proceedings of the 10th ACM International Conference on Computer and Communications Security*, pages 8–19, 2003.
- [48] Aristides Mpitzopoulos, Damianos Gavalas, Charalampos Konstantopoulos, and Grammati Pantziou. A Survey on Jamming Attacks and Countermeasures in WSNs. *IEEE Communications Surveys & Tutorials*, 11(4):42–56, 2009.
- [49] Vishnu Navda, Aniruddha Bohra, Samrat Ganguly, and Dan Rubenstein. Using Channel Hopping to Increase 802.11 Resilience to Jamming Attacks. In *Proceedings of the International Conference on Computer Communications (INFOCOM)*, pages 2526–2530, 2007.
- [50] Dragoş Niculescu. Interference Map for 802.11 Networks. In *Proceedings of the 7th ACM SIGCOMM Internet Measurement Conference (IMC)*, pages 339–350, 2007.
- [51] Noam Nisan, Time Roughgarden, Éva Tardos, and Vijay V. Vazirani. *Algorithmic Game Theory*. Cambridge University Press, 2007.
- [52] Andrzej Pelc and David Peleg. Broadcasting with Locally Bounded Byzantine Faults. *Information Processing Letters*, 93(3):109–115, 2005.
- [53] Andrzej Pelc and David Peleg. Feasibility and complexity of broadcasting with random transmission failures. In *Proceedings of the 24th Symposium on Principles of Distributed Computing*, pages 334–341, 2005.
- [54] Konstantinos Pelechrinis, Marios Iliofotou, and Srikanth V. Krishnamurthy. Denial of Service Attacks in Wireless Networks: The Case of Jammers. *IEEE Communications Surveys & Tutorials*, 13(2):245–257, 2011.
- [55] Joseph Polastre, Robert Szewczyk, and David Culler. Telos: Enabling Ultra-Low Power Wireless Research. In *Proceedings of the 4th International Symposium on Information Processing in Sensor Networks*, 2005.
- [56] Prolexic Technologies, Inc. www.prolexic.com.

- [57] Iyappan Ramachandran and Sumt Roy. Clear Channel Assessment in Energy-Constrained Wideband Wireless Networks. *IEEE Wireless Communications*, 14(3):70–78, 2007.
- [58] Andrea Richa, Christian Scheideler, Stefan Schmid, and Jin Zhang. A Jamming-Resistant MAC Protocol for Multi-Hop Wireless Networks. In *Proceedings of the International Symposium on Distributed Computing (DISC)*, 2010.
- [59] Sara Robinson. The Price of Anarchy. *SIAM News*, 37(5):1–4, 2004.
- [60] Vyas Sekar and Jacobus Van Der Merwe. LADS: Large-scale Automated DDoS Detection System. In *Proceedings of the USENIX ATC*, pages 171–184, 2006.
- [61] Micah Sherr, Michael Greenwald, Carl A. Gunter, Sanjeev Khanna, and Santosh S. Venkatesh. Mitigating DoS Attack Through Selective Bin Verification. In *Proceedings of the First international conference on Secure network protocols*, NPSEC’05, pages 7–12, 2005.
- [62] D. Sleator and R. Tarjan. Amortized Efficiency of List Update and Paging Rules. *Communications of the ACM*, 28(2):202–208, 1985.
- [63] Dongjin Son, Bhaskar Krishnamachari, and John Heidemann. Experimental study of the effects of Transmission Power Control and Blacklisting in Wireless Sensor Networks. In *Proceedings of the 1st IEEE Conference on Sensor and Adhoc Communication and Networks*, pages 289–298, Santa Clara, California, USA, October 2004. IEEE.
- [64] Dongjin Son, Bhaskar Krishnamachari, and John Heidemann. Experimental Study of Concurrent Transmission in Wireless Sensor Networks. In *Proceedings of the 4th International Conference on Embedded Networked Sensor Systems, SenSys*, pages 237–250, 2006.
- [65] Kannan Srinivasan and Philip Levis. RSSI is Under Appreciated. In *Proceedings of the 3rd Workshop on Embedded Networked Sensors (EmNets)*, 2006.
- [66] James Sundali and Rachel Croson. Biases in Casino Betting: The Hot Hand and the Gamblers Fallacy. *Judgment and Decision Making*, 1(1):1–12, 2006.
- [67] Vinod Vaikuntanathan. Brief announcement: Broadcast in Radio Networks in the Presence of Byzantine Adversaries. In *Proceedings of the 24th Annual ACM symposium on Principles of Distributed Computing*, pages 167–167, 2005.
- [68] Ashlee Vance. WikiLeaks Struggles to Stay Online After Attacks. www.nytimes.com/2010/12/04/world/europe/04domain.html?_r=2&hp, 2010.
- [69] Michael Walfish, Mythili Vutukuru, Hari Balakrishnan, David Karger, and Scott Shenker. DDoS Defense by Offense. In *Proceedings of the 2006 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (SIGCOMM)*, pages 303–314, 2006.
- [70] John Paul Walters, Zhengqiang Liang, Weisong Shi, and Vipin Chaudhary. *Security in Distributed, Grid, Mobile, and Pervasive Computing. Chapter 17: Wireless Sensor Network Security: A Survey*. Auerbach Publications, 2007.
- [71] Ronald Watro, Derrick Kong, Sue fen Cuti, Charles Gariner, Charles Lynn, and Peter Kruus. TinyPK: Securing Sensor Networks with Public Key Technology. In *Proceedings of the 2nd ACM Workshop on Security of Ad Hoc and Sensor Networks (SASN)*, pages 59–64, 2004.

- [72] Anthony D. Wood and John A. Stankovic. Denial of Service in Sensor Networks. *Computer*, 35(10):54–62, 2002.
- [73] Wenyuan Xu, Ke Ma, Wade Trappe, and Yanyong Zhang. Jamming Sensor Networks: Attack and Defense Strategies. *IEEE Network*, 20(3):41–47, 2006.
- [74] Wenyuan Xu, Wade Trappe, Yanyong Zhang, and Timothy Wood. The Feasibility of Launching and Detecting Jamming Attacks in Wireless Networks. In *Proceedings of the 6th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*, pages 46–57, 2005.
- [75] Wei Ye, John Heidemann, and Deborah Estrin. An Energy-Efficient MAC Protocol for Wireless Sensor Networks. In *Proceedings of the 21st International Conference on Computer Communications (INFOCOM)*, pages 1567–1576, 2002.
- [76] Maxwell Young and Raouf Boutaba. Overcoming Adversaries in Sensor Networks: A Survey of Theoretical Models and Algorithmic Approaches for Tolerating Malicious Interference. Accepted to *IEEE Communications Surveys & Tutorials*, 2011.
- [77] Marco Zuniga and Bhaskar Krishnamachari. Analyzing the Transitional Region in Low Power Wireless Links. In *Proceedings of the 1st IEEE International Conference on Sensor and Ad hoc Communications and Networks (SECON)*, pages 517–526, 2004.